SUITE BRITEL

IOSU SAN MARTÍN GONZÁLEZ

MÁSTER EN INGENIERÍA INFORMÁTICA, FACULTAD DE INFORMÁTICA,
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin de Máster en Ingeniería Informática

1 de septiembre de 2018

Director: Adrián Riesco Rodríguez

Autorización de Difusión

IOSU SAN MARTÍN GONZÁLEZ

1 de septiembre de 2018

El abajo firmante, matriculado en el Máster en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "SUITE BRITEL", realizado durante el curso académico 2017-2018 bajo la dirección de Adrián Riesco Rodríguez en el Departamento de Sistemas Informáticos y Computación, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Resumen en castellano

La información, ese bien que influencia tiene en el mundo moderno por su capacidad de provocar cambios en quién la recibe, alcanza su máximo exponente cuando se produce su difusión. Uno de los canales más importantes de difusión y transmisión de la información es la TV, que utiliza diferentes medios de transmisión (telecomunicaciones) para hacer llegar información (audio y video) desde una estación emisora hasta el usuario final.

Estos medios de transmisión que aportan los proveedores de servicios en el área de las telecomunicaciones, debido a que están sujetos a un proceso permanente de innovación tecnológica (materiales, nuevos equipos…), están generando un despliegue de redes de nueva generación (ultrarrápidas), lo que está provocando un importante impacto en el sector de la TV y en toda la sociedad en general.

Esta situación de evolución tecnológica y revolución social, que esta provocando un gran cambio en las relaciones interpersonales, no puede pasar desapercibida para el Ingeniero Informático, pieza importante en el tratamiento de la Información y el mantenimiento de los Sistemas de Telecomunicaciones. Por ello, en este Trabajo Fin de Máster se ha diseñado y desarrollado una aplicación informática que permite gestionar todo el "sistema" de la empresa de distribución de canales de Televisión "Britel S.A.", mediante la creación de una página web que sirva para la introducción y mantenimiento de datos por parte del administrador del sistema y por otro lado que permita al usuario introducir sus opciones y preferencias elegidas.

Esta aplicación API se denomina "SUITE BRITEL", porque en ella se gestionan todas las aplicaciones y servicios que ofrece la empresa, y el autor pretende que sea sencilla, cumpla su función de abstracción con el resto de aplicaciones y que sirva de forma eficaz y potente a la gestión de los recursos de la empresa. Es interesante notar la componente de innovación de este trabajo: dotar a la API de flexibilidad orientada al cliente y usuario de la información, con el fin de que este se sienta consumidor activo en el proceso de comunicación.

Palabras clave

- SEÑAL DE VIDEO.
- DIGITALIZACION (Informática)
- IPTV
- TECNOLOGIAS DE DIFUSION (broadcast, geocast, streaming)
- SET TOP BOX
- BRITEL S.A.
- HIPER-INFORMACION (Incertidumbre)
- API (*Application Programming Interface*)

Resumen en inglés

Information, that asset that so much influence has in the modern world because of its ability to trigger changes in those who receives it, reaches its maximum expression when it is distributed. One of the most important channels for the dissemination and transmission of information is TV, which uses different media (telecommunications) to deliver information (audio and video) from a broadcast station to the end user.

These media of transmission offered by service providers in the area of telecommunications due to their subjection to a permanent process of technological innovation (materials, new equipment...), are generating a deployment of new generation networks (ultra-fast), which is causing a significant impact on the TV sector and society in general.

This technological evolution and social revolution, which is causing a great change in interpersonal relations, cannot be ignored by the Computer Engineer, an important part in the treatment of Information and the maintenance of Telecommunications Systems. For this reason, in this Master's Thesis, a computer application has been designed and developed to manage the entire "system" of the television channel distribution company "Britel S.A.", by creating a web page that is used for the introduction and maintenance of data by the system administrator and, on the other hand, allowing the user to insert his or her chosen options and preferences.

This API application is called "Suite Britel" because it manages all the applications and services offered by the company, and the author wants it to be simple, to fulfill its function of abstraction with the rest of applications, and to be efficient and powerful in the management of the company's resources. It is interesting to note the innovation component of this work: to provide the API with flexibility oriented to the client and user of the information, so that they feel that they are active consumers in the communication process.

Keywords

- VIDEO SIGNAL
- DIGITALIZATION
- IPTV
- DIFFUSION TECHNOLOGIES
- SET TOP BOX
- BRITEL S.A.
- HYPERINFORMATION
- WEP API

Índice

Capítulo 1 - Introducción

En este primer capítulo se va a hacer una introducción de este Trabajo de Fin de Master. Para ello, empezaremos presentando una breve motivación y una justificación de la necesidad del trabajo, continuaremos con los objetivos planteados y concluiremos con un resumen de los contenidos de la memoria.

Justificación

Si se toma la definición de Información como "dato o conjunto de datos con capacidad de ser procesados (adquisición, almacenamiento y tratamiento) y cuyo resultado provoca un cambio en el entorno (conocimiento)", nos encontramos ante un "bien material" de carácter volátil, que adquiere un "valor" en función de la utilidad que tenga esa información para el destinatario y por tanto es susceptible de ser difundido a través de los medios de comunicación.

Para que los datos puedan ser procesados y difundidos informáticamente, es necesario que estos sean susceptibles de ser representados mediante una señal analógica y/o digital. Es en este punto donde encontramos las principales fuentes de información: el sonido, la imagen y el texto, fundamento de las principales tecnologías de las ciencias de la Información y las Telecomunicaciones como son: la Radio, la Televisión y la Telefonía.

El "boom" que las nuevas tecnologías están provocando en los procesadores de información y en la velocidad de transmisión mediante las redes de telecomunicaciones, está originando un gran impacto en toda la sociedad. Hoy en día la información constituye un factor social, económico y cultural de gran relevancia, sobre todo la transmitida mediante los canales de TV y la Telefonía.

En este contexto es donde aparece la importancia de la Informática: para que todo el sistema de las Telecomunicaciones pueda funcionar es necesario dotarlo de inteligencia, mediante la creación de programas y protocolos que mantengan todo el sistema y aceleren la "necesaria adaptación" a este vertiginoso cambio que están generando las nuevas tecnologías y que están provocando una enorme expansión de Internet y un desarrollo acelerado de su utilización.

Para alcanzar ese equilibrio y adaptación a los nuevos cambios que se están produciendo y seguirán produciéndose en el futuro, como Ingeniero Informático es mi deseo contribuir y colaborar en la creación de arquitecturas más inteligentes de las redes de comunicación y en los equipos periféricos de la red (terminales), aportando una mayor eficacia y mejores resultados en cualquiera de los sectores de la Sociedad.

Motivación

Dentro de mis estudios de Máster en Ingeniería Informática la Universidad Complutense de Madrid me ha dado la oportunidad de realizar prácticas laborales en una empresa del sector de los medios de comunicación Audiovisuales. Esta empresa, Britel S.A., se dedica a la distribución de canales de TV mediante el uso de protocolos de comunicación y en este sentido ofrece el servicio "IPTV", televisión basada en el protocolo "IP", un servicio muy atractivo para los usuarios finales y que permite a las empresas competir y posicionarse en el mercado.

La experiencia obtenida durante mis prácticas laborales en la empresa Britel S.A. me ha permitido establecer que, debido a la alta competitividad existente en el área de las comunicaciones desde un punto de vista comercial, la empresa en su objetivo de optimizar resultados para la consecución de mayores beneficios orienta todas sus energías en proveer a sus clientes de una mayor cantidad y mejores accesos a los medios audiovisuales, todo ello en perjuicio de otros objetivos no menos importantes, como la optimización de los recursos software y la interacción con sus clientes.

Esta situación lleva a una deshumanización del proceso de Información y Comunicación que realiza la empresa Britel S.A., algo que es extensible al resto de la sociedad. En la actualidad, tanto los trabajadores como los clientes de la empresa son protagonistas pasivos de este consumo de información que invade todo el mercado.

Sin pretender mantenerme ajeno al mundo comercial, cuyo objetivo fundamental es la obtención de beneficios, y en la creencia que no está reñido con un mejor servicio al cliente, este trabajo Fin de Máster pretende optimizar el rendimiento de los trabajadores de la empresa mediante un mejor aprovechamiento de los recursos software y un acercamiento mayor al usuario final de la Información, humanizando el proceso de distribución y acceso a los canales de TV ofrecidos por la empresa Britel S.A., facilitando la toma de decisiones a sus clientes, mediante la elección de sus preferencias.

En la actualidad, los clientes solo pueden optar a todos los canales ofrecidos por Britel S.A., no existe la posibilidad de elección de canales y por lo tanto tampoco es posible diferenciar la tarifación.

Por este motivo y como complemento de las aplicaciones ya existentes en la empresa, en este trabajo se ha diseñado una API y un entorno *web* que facilite el mantenimiento de todo el sistema de la empresa y la interacción con los clientes. Este proyecto ha sido aprobado por el jefe del departamento de TI donde actualmente estoy trabajando, y se implantará al finalizar su desarrollo en caso de obtener resultados positivos.

Objetivos

Para la realización de este Trabajo Fin de Máster se establecen los siguientes objetivos:

General:

Diseño y desarrollo de una aplicación (API) que gestiona el sistema de una empresa que comercializa la distribución de canales TV mediante la plataforma IPTV. Para ello se desarrollará una aplicación WEB que permita la introducción de datos para el mantenimiento del sistema, por parte de los siguientes actores:

* Administrador → mantenimiento del sistema.
* Operador → mantenimiento de las bases de datos
* Cliente / usuario → introduce preferencias en base a las opciones ofertadas.

Específicos:

1- Estudio de los sistemas y arquitecturas de redes utilizadas en la actualidad para la retransmisión de IPTV.
2- Identificar y estudiar cada uno de los componentes (hardware) que conforman los sistemas IPTV y describir sus características principales.
3- Estudio de las tecnologías utilizadas por los medios de trasmisión para la difusión de la información a los terminales periféricos (usuarios).
4- Estudio de los protocolos de comunicación utilizados en los sistemas IPTV.
5- Estudio de los lenguajes de programación Java Spark (API), SQL (Base De Datos), HTML5 (página web) y Java Script (página web), utilizados para desarrollar la

aplicación que gestiona el sistema de la empresa de distribución de canales TV mediante la plataforma IPTV.

Estructura de la memoria

A continuación, se describen los capítulos que conforman este Trabajo Fin de Máster:

- Capítulo 1: en este capítulo se define el principal objeto de este trabajo y que consiste en el diseño y desarrollo de una API orientada a gestionar el sistema de una empresa dedicada a la venta de canales de TV mediante Internet (IPTV) y su interacción con los clientes. En primer lugar, se realiza una introducción a los conceptos que justifican la intervención de la Informática en la creación y mantenimiento de todo proceso de comunicación, para continuar con los antecedentes y hechos que motivan al autor a desarrollar este trabajo y finalmente se mencionan las diferentes materias a estudio que han sido necesarias para llegar al objetivo principal.

- Capítulo 2: En este capítulo se realiza un resumen sobre la legislación y normativa aplicable vigente, necesaria para la "seguridad y privacidad" de la Información y que ha de tener en cuenta el programador para el tratamiento de los datos personales en el ámbito de las Telecomunicaciones y en la difusión de Televisión. También en este capítulo se hace referencia a las tecnologías emergentes y de nueva generación que hacen posible los sistemas y arquitecturas de redes utilizadas para la retransmisión de IPTV, sus componentes (hardware y protocolos) y por último una breve referencia a los Leguajes de programación utilizados por el autor para el desarrollo de este trabajo.

- Capítulo 3: En este capítulo se establece la metodología de trabajo. En primer lugar, se especifican los requisitos (arquitectura) del sistema que van a formar parte de "Suite Britel" y las tecnologías y lenguajes utilizados para el diseño y desarrollo de la API y la página web. A continuación, se realiza un estudio de las funcionalidades que requiere el sistema, al objeto de determinar los actores que van a participar y para ello se utilizan de forma detallada "diagramas de casos de uso". Finalmente se hace un repaso de todo aquello que va a ser necesario aportar al trabajo y se concluye con un cronograma de implementación.

- Capítulo 4: En este capítulo se presenta en detalle el desarrollo de las principales funciones que utiliza la API para conseguir el objetivo general y su implementación en la empresa Britel S.A. Así mismo, se presenta el desarrollo de una web básica al objeto de visualizar el funcionamiento de la API y la estructura de las bases de datos utilizadas.

- Capítulo 5: En este capítulo se describen las pruebas realizadas y los resultados obtenidos durante la implementación del trabajo y se hace una referencia a las observaciones y experiencias desarrolladas de los aspectos prácticos del sistema implementado.

- Capítulo 6: En este capítulo se hace entrega de las reflexiones y conclusiones acerca del trabajo fin de máster desarrollado y se establecen los aportes de mejora obtenidos con la implantación de la API en el sistema de trabajo de la empresa Britel S.A.

- Apéndices: Al final del trabajo, como apéndice se adjuntan la explicación para instalar el entorno del trabajo y los dos protocolos utilizados de comunicación entre la API, la página web y los dispositivos

El código fuente de la aplicación se encuentra disponible en GitHub y es totalmente accesible para toda la comunidad, a través del siguiente enlace: https://github.com/iosusam/Trabajo-Fin-De-Master

Capítulo 1 - Introduction

This first chapter introduces the rest of this Master's thesis. We will start by presenting a brief motivation and justification of the need for the work, continue with the objectives established, and conclude with a summary of the contents of the report.

Justification

If we take the definition of Information as "data or a set of data with the capacity to be processed (acquisition, storage, and processing) and the result of which causes a change in the environment (knowledge)", we are faced with a "material good" of a volatile nature, which acquires a "value" depending on the usefulness of this information for the recipient and which can be diffused through the media.

In order for the data to be processed and diffused electronically, it must be capable of being represented by an analogue and/or digital signal. It is at this point where we find the main sources of information: sound, image, and text, the basis of the main technologies of the Information and Telecommunications Sciences such as: Radio, Television and Telephony.

The "boom" that new technologies are causing in information processors and in the speed of transmission through telecommunications networks is having a great impact on society as a whole. Nowadays, information is a very important social, economic, and cultural factor, especially the one transmitted through TV channels and Telephony.

It is in this context that the importance of Information Technologies appears: in order for the whole Telecommunications system to work, it is necessary to provide it with intelligence, through the development of programs and protocols that maintain the whole system and accelerate the "necessary adaptation" to this vertiginous change that new technologies are generating and that are causing an enormous expansion of the Internet and an accelerated development of its use.

In order to achieve this balance and to adapt to the new changes that are taking place and will continue to take place in the future, as a Computer Engineer I wish to contribute and collaborate in the creation of more intelligent architectures of the communication networks and network equipment (terminals), providing more efficiency and better results in all sectors of Society.

Motivation

As part of my Master's Degree in Computer Engineering, Complutense University of Madrid has given me the opportunity to do an internship in a company in the Audiovisual media sector. This company, Britel S.A., is dedicated to the distribution of TV channels through the use of communication protocols and offers the "IPTV" service and television based on the "IP" protocol, a very attractive service for end users that allows companies to compete and position themselves in the market.

The experience obtained during my internship at Britel S.A. has allowed me to establish that, due to the high competitiveness in the area of communications from a commercial point of view, the company, in its objective of optimizing results in order to achieve greater benefits, focuses all its energies on providing its clients with a greater quantity and better access to audiovisual media. All of this is not less important than other objectives, such as the optimization of software resources and the interaction with its clients.

This situation leads to a dehumanization of the Information and Communication process performed by this the company, something that can be extended to the rest of society. Today, the company's employees and customers are both passive players in this consumption of information that invades the entire market.

Trying not to stay out from the commercial field, whose fundamental objective is to obtain benefits, and in the belief that it is not at odds with better customer service, this Master's thesis aims to optimize the performance of the company's employees through a better use of software resources and a greater approach to the end user of information. Humanizing the process of distribution and access to TV channels offered by Britel SA and facilitating decision-making for its customers by choosing their preferences.

At present, customers can only choose all the channels offered by the previous mentioned company, there is no choice of channels individually and therefore it is not possible to differentiate the price.

For this reason, and as a complement to the existing applications in the company, in this work I have designed an API and a web environment that facilitates the maintenance of the entire system of the company and interaction with customers.

Objective

The following objectives are established for the completion of this Master's Degree:

General:

Design and development of an application (API), which manages the system of a company that offers the distribution of TV channels through the IPTV platform. To this end, a WEB application will be developed to allow the input of data for the maintenance of the system by the following actors:

* Administrator → system management.
* Operator → database management.
* Client / user → enters preferences based on the options offered.

Specific:

1- Study of the systems and network architectures currently used for IPTV retransmission.
2- Identify and study each of the components (hardware) that make up the IPTV systems and describe their main characteristics.
3- Study of the technologies used by transmission media for the diffusion of information to terminals (users).
4- Study of the communication protocols used in IPTV systems.
5- Study of the Java Spark (API), SQL (Database), HTML5 (web page) and Java Script (web page) programming languages, used to develop the application that manages the TV channel distribution company's system using the IPTV platform.

Structure of this Master's thesis

The chapters that define this Master's thesis are described below:

- Chapter 1: This chapter defines the main objective of this work, which consists of the design and development of an API aimed at managing the system of a company dedicated to the sale of Internet TV channels (IPTV) and their

interaction with customers. First, an introduction is made to the concepts that justify the intervention of Information Technology in the creation and maintenance of all communication processes, in order to continue with the background and facts that motivate the author to develop this work and finally the different subjects that have been necessary to reach the main objective are mentioned.

- Chapter 2: This chapter provides a summary of the applicable legislation and regulations in use, necessary for the "security and privacy" of the Information and to be taken into account by the programmer for the processing of personal data in the area of Telecommunications and Television diffusion. This chapter also refers to the emerging and new generation technologies that make possible the systems and network architectures used for IPTV broadcasting, its components (hardware and protocols) and, finally a short reference to the programming languages used by the author for the development of this work.

- Chapter 3: This chapter sets out the working methodology. First, the system requirements (architecture) to be part of the “Suite Britel” and the technologies and languages used for the design and development of the API and website are specified. A study of the functionalities required by the system is then carried out to determine the actors to be involved, using detailed "use case diagrams". Finally, a review is made of everything that is going to be necessary to support the work and the implementation schedule is concluded.

- Chapter 4: In this chapter we present in detail the development of the main functions used by API to achieve the general objective and its implementation in Britel S.A. We also present the development of a basic website in order to visualize functionality of API and structure of the databases used.

- Chapter 5: This chapter describes the tests performed and the results obtained during the implementation of the work and refers to the observations and experiences developed on the practical aspects of the system implemented.

- Chapter 6: This chapter presents the reflections and conclusions on the final work of the master's degree and establishes the contributions to improvement obtained with the implementation of API in the work system of the company Britel S.A.

- Appendices: At the end of the work, the explanation for installing the work environment and the two communication protocols used between API, Web Page, and the devices are attached as an appendix.

The application's source code is available from GitHub and is fully accessible to the entire community through the following link: https://github.com/iosusam/Trabajo-Fin-De-Master

Capítulo 2 - Preliminares

En este capítulo se va a realizar una descripción de los fundamentos y tecnologías utilizadas en la distribución de la señal de televisión, desde su captación en origen hasta que llega al usuario final, así como una referencia a los lenguajes de programación utilizados en el desarrollo de una API que gestiona la distribución de canales de televisión por parte de la empresa Britel S.A.

En primer lugar, empezaremos explicando los conceptos básicos de la Televisión Digital y profundizando en las diferentes etapas por las que pasa la señal de video desde su digitalización hasta su difusión y recepción. A continuación, explicaremos los diferentes sistemas de difusión que existen, como su acceso para visualizar la televisión digital, y diferentes tecnologías y herramientas utilizadas para el desarrollo de este proyecto. Concluiremos con las normas aplicables en Europa para estos entornos.

La televisión

La televisión, cuyo significado etimológico es visión a distancia, "es un sistema de comunicación que consiste en un intercambio de mensajes de voz e imagen a una cierta distancia" [Capítulo 0 - Appendix -].

Desde el punto de vista de la Teoría de las Telecomunicaciones se trata de "un sistema de transmisión/recepción de señales (imágenes en movimiento, sonido y texto), mediante un mecanismo de difusión (alteración del medio), ya sean ondas hertzianas o impulsos eléctricos o luminosos" [0].

La difusión de la televisión se compone de varias etapas como se puede observar en la figura 1. En la parte de la izquierda, se puede observar la captación de la imagen y su conversión a señal de video. Continuando con la parte central, se observa la modulación y la transmisión de dicha señal de video y en la parte derecha finalizando con la recepción y amplificación de esta señal de video para su visualización.

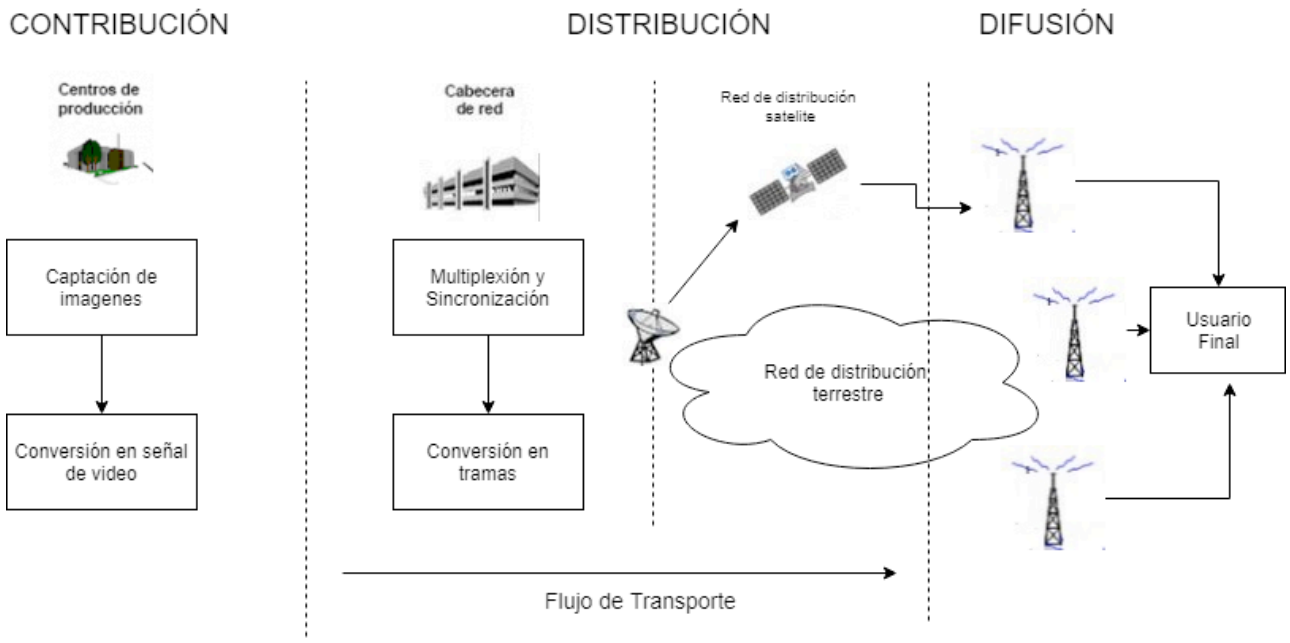


Figura 1: Etapas de difusión de televisión

A continuación, se va a explicar brevemente las diferentes fases y su relación con la empresa Britel S.A. para entender el origen de los canales, que la API desarrollada va a utilizar como fuente de información y la transmitirá hasta los dispositivos de los clientes

La señal de video

La primera fase de la televisión es la captación de imágenes y sonido para que puedan ser enviadas a distancia. La empresa Britel S.A. realiza esta primera fase mediante el departamento Marketing & Digitalización, que se encuentra en los "centros de producción", produciendo contenido de imagen y video propio (Ver figura 1, lateral izquierdo). Este contenido, será transformado en la siguiente etapa que se explica en el siguiente apartado. La realización de este contenido, tiene como consecuencia la creación de canales propios por internet como: canal Ñam (de cocina), canal Pequeñines (dibujos animados), canal 80s (música), …

Según viene recogido en la Teoría de la Señal [0], tanto la imagen como el sonido y el texto son fuentes de información susceptibles de ser convertidas en una señal para su transmisión a través de algún medio de difusión. Por tanto, se define señal como "un flujo de información proveniente de una fuente y que puede tener diferente naturaleza: mecánica, eléctrica,

acústica…" Por lo general la información se transforma a señales eléctricas para que pueda ser procesada.

La señal de video [0] es una señal eléctrica variable que contiene diferentes tensiones (voltaje) dependiendo de la luminosidad de la imagen a trasmitir. Se origina mediante la conversión de variaciones de intensidad luminosa (obtenidas de la imagen) a variaciones de intensidad eléctrica.

La señal de video está compuesta por dos tipos de información:

1- Luminancia [0]: contiene las variaciones de intensidad luminosa que sirven para representar cada punto de la imagen en blanco y negro (escala de grises).

2- Crominancia [0]: contiene información sobre los colores rojo, verde y azul y representa la imagen mediante la combinación de los tres colores primarios.

Existen dos tipos de señal para poder transmitir información con eficiencia que se explican a continuación:

- **Señal analógica (eléctrica)** [0]**:**

  Las variaciones de tensión eléctrica pueden tomar valores entre 0 y 1 por lo tanto, puede tomar infinitos valores intermedios y se comporta como una variable continua que codifica la información sobre las imágenes de forma analógica. La gran desventaja que supone este tipo de señal son las degradaciones que sufre el contenido al realizar copias y la difícil recuperación que supone en caso de una transmisión fallida. Es por estos dos motivos por lo que en la empresa se utiliza la señal digital que se explica a continuación.

- **Señal Digital (dato informático)** [0]**:**

  La digitalización de la señal de video consiste en el tratamiento de las imágenes de forma discreta, mediante la codificación de la información con "0s y 1s" (BIT), lo que facilita el almacenamiento de la señal de video en servidores y archivos para su posterior tratamiento (compresión, inclusión de "metadatos", difusión) y otras aplicaciones.

En producción de Televisión, la señal de audio y texto son tratadas por separado y posteriormente trasmitidas junto con la señal de video para su emisión.

Modulación / Transmisión

La empresa Britel S.A. ha producido información y contenido, explicado en el apartado anterior, sobre los canales de televisión que quiere transmitir, por lo que a continuación explicaremos brevemente las técnicas que existen en el mercado para poder transmitir y modular el contenido para posteriormente ser visualizado.

En primer lugar, definiremos los conceptos básicos para entender este apartado. Se denomina modulación al "conjunto de técnicas que se utilizan para transportar la información que contiene una señal (imagen, sonido o texto) hasta otro lugar que se encuentra a una cierta distancia y separado por un medio capaz de transmitir ondas" [0]. Este transporte de la señal se realiza a través de una onda portadora, normalmente ondas hertzianas (sinusoidales).

Una **señal portadora** es "una onda electromagnética susceptible de ser modificada en alguno de sus características por la **señal de información** (video, sonora o texto) al objeto de obtener una **señal modulada** que realmente es la señal que transporta la información a través del canal o medio de comunicación" [0].

Por tanto, la modulación consiste en hacer variar un parámetro de la onda portadora de acuerdo con las variaciones de la **señal moduladora** (señal de información), que es la que realmente deseamos transmitir.

Esta técnica de modulación (compresión) utilizada, en función del parámetro sobre el que se actúe para variar la señal portadora, determina los diferentes tipos de modulación, siendo tres los más básicos y sobre todo los más utilizados por la mayoría de los sistemas de comunicación:

- Modulación de la amplitud (AM).
- Modulación de la frecuencia (FM).
- Modulación de fase (PM).

El resultado de todo este proceso es una señal que puede ser modulada sobre una onda portadora de radiofrecuencia y transmitida a través de una antena.

La empresa dispone de varias antenas por el territorio nacional para poder transmitir los contenidos que produce y enviarlos a la sede principal que se encuentra en Madrid.

Recepción / Visualización

El desarrollo de este proyecto se ubica en este apartado, debido a que los dispositivos que se encargan de la recepción y la visualización de los canales de televisión que difunde Britel S.A., deben de conocer la localización donde se encuentra esta información. Además, la localización de los canales sufre modificaciones, las cuales son necesarias tratarlas para comunicárselo a los dispositivos y puedan recepcionar la señal.

En relación al párrafo anterior, se expone las definiciones básicas para entender su funcionamiento. Al proceso de recuperar la información transmitida por las ondas portadoras se le llama demodulación y consiste en realizar el proceso inverso, al objeto de poder visualizar las imágenes en movimiento, tal y como se encontraban antes de la codificación y sincronizarlas con el sonido también transmitido.

Una vez transmitida la señal modulada es necesario que sea recuperada por algún dispositivo receptor de señales de TV y posteriormente demodulada/decodificada para su visualización.

Se denomina **Televisor** al aparato electrónico destinado a la recepción de la señal de Televisión. Está formado por un sintonizador y por mandos y circuitos necesarios para la conversión de las señales eléctricas (analógicas o digitales) en representaciones de imágenes en movimiento en una pantalla y el sonido a través de altavoces.

Se denomina Decodificador o Receptor de TV (Set Top Box) al dispositivo electrónico que al igual que el Televisor recepta y decodifica las señales de TV (analógicas o digitales) al objeto de que puedan ser visualizadas en cualquier otro dispositivo que disponga de pantalla.

Televisión digital / Digitalización

La televisión digital es el medio de comunicación a distancia que transmite, recibe y procesa señales de audio y video, que previamente han sido digitalizadas (codificadas mediante 1s y 0s).

Se entiende por digitalización [0] al conjunto de técnicas que permiten la conversión/transcripción de señales analógicas en señales digitales con el objetivo de facilitar su

procesamiento (codificación, compresión, …) y transporte a través de las redes de telecomunicaciones.

En la oficina de Madrid de Britel S.A., la cual recibe todas las señales analógicas enviadas desde diferentes puntos del territorio español, se encuentran varios servidores que se encargan de la transformación mediante los siguientes procesos:

1- Muestreo:

   Consiste en tomar muestras de la amplitud de una señal analógica (tensión) a una determinada frecuencia. Todavía se trata de una variable continua que puede tomar infinitos valores y por tanto analógica.

2- Cuantificación:

   Consiste en comprimir un rango de valores conseguido durante el muestreo en un único valor y con ello se consigue que se comporte como una variable discreta preparada para ser codificada. Para ello se establece un determinado umbral, los valores que están por encima toman un valor y los que están por debajo otro valor.

3- Codificación digital:

   Es el último de los procesos y consiste en traducir esos valores de tensión eléctrica obtenidos al sistema binario (sucesión de unos y ceros).

Una de las desventajas de la digitalización es la cantidad de datos que se generan, teniendo en cuenta que la capacidad de almacenamiento de los soportes es limitada y que los equipos de transmisión solo pueden manejar una determinada tasa de datos, se hace necesario añadir otro proceso a la digitalización:

- Compresión:

  Se trata de un caso particular de la codificación y consiste en la reducción de datos mediante la aplicación de algoritmos matemáticos.

  Esta fase, es realizada por un software de la empresa que se adquirió mediante un proveedor externo.

La digitalización en la Televisión tiene dos partes bien diferenciadas: por un lado, la producción de señales de TV y por otro lado la transmisión de la señal a través de los medios de Telecomunicaciones.

La televisión digital está orientada principalmente a la transmisión de las señales y la digitalización ha permitido la utilización de diferentes tecnologías que facilitan la transmisión de la señal de manera más rápida y eficiente, las cuales veremos en el siguiente apartado. Algunas de las ventajas de la Televisión digital son:

1- Permite la utilización de técnicas de compresión.
2- Mayor aprovechamiento del ancho de banda, mediante un uso más eficiente del espectro radioeléctrico permitiendo transmitir mediante la técnica de multiplexación más de una señal televisiva. Al espacio antes ocupado por una sola señal y por el cual ahora es aprovechado para el envío de más señales se le llama canal múltiple digital (MULTIPLEX).
3- La cantidad de información transmitida por un canal MULTIPLEX dependerá de la relación de compresión empleada.
4- Posibilidad de incluir texto y datos adicionales que aporten información (metadatos).
5- Audio y video de calidad superior.
6- Mejoras en las tecnologías de visualización. Superior resolución.
7- Convergencia TV-PC (servicios de Internet).

Tecnologías de transmisión / Flujo de transporte digital

Para que la señal digital de TV pueda ser transportada de un lugar a otro distante del primero, es necesaria la aplicación de diferentes tecnologías que sirvan de enlace entre los dos lugares y que permitan reproducir la señal transmitida, tal y como fue captada en el lugar de origen. Es por ello, que la empresa ha tenido que aplicar una tecnología para el transporte de la señal.

1- Las diferentes tecnologías de transporte de la señal digital de TV que existen, se pueden agrupar en dos tipos:

1- Tecnologías de codificación–decodificación:

Permiten procesar la señal durante todo el proceso de transmisión, para que llegue hasta el usuario final en las mejores condiciones de reproducción.

Estas tecnologías se basan en el flujo de transporte MPEG (Moving Picture Experts Group), organismo encargado de crear y proponer los procedimientos de estandarización (normas) con el objetivo de crear una televisión digital compatible:

- Establece los aspectos y metodologías de compresión de las señales de audio y video.

- Establece los procedimientos de multiplexación y sincronización de las señales de TV, mediante la creación de tramas de programa y transporte.

2- Tecnologías de telecomunicaciones:

Permiten transportar la señal a distancia mediante el uso de los sistemas de modulación y acceder a ella mediante la tecnología DVB (Digital Video Broadcast).

El DVB es un organismo encargado de crear y proponer los procedimientos de estandarización (normas) que definen los sistemas de modulación de señal, necesarios para el transporte de las tramas creadas mediante la tecnología MPEG, en función del tipo de radiodifusión: satelital, cableada o terrestre.

Broadcast es una tecnología de difusión que consiste en transmitir la información a través de la red, independientemente de haber sido solicitada. Esta información difundida está a disposición de cualquier dispositivo que se conecte a la red.

Esta tecnología es por la que ha apostado la empresa debido a que sus usuarios finales reciban la información de los canales mediante dispositivos conectados a la red.

Sistemas de difusión y acceso a la televisión digital

La señal de Televisión durante el proceso de transmisión / distribución pasa por diferentes procesos desde su captación / producción hasta que llega al receptor (televisor o decodificador) del usuario final y es en este último punto, donde la transmisión de la señal al usuario final se realiza de forma totalmente digital mediante diferentes tecnologías de acceso.

Todos los sistemas de transmisión digital tienen un esquema parecido y las diferencias fundamentales son:

- El formato de audio y video utilizado antes y después de la codificación.
- El sistema de modulación empleado en función del medio de difusión utilizado.
- Como el flujo de transporte se convierte en señal de emisión.

Televisión digital Terrestre

"Es la transmisión de imágenes en movimiento y su sonido asociado mediante señales digitales (codificación binaria) a través de una red de repetidores terrestres que utilizan las ondas hercianas" [0]. Estas ondas utilizan la atmósfera para trasladar la información (espectro radioeléctrico). Las principales características son:

1- La TDT opera en la banda UHF, que ha sido utilizada por la TV analógica.

2- La TV analógica solo permitía un único programa por canal UHF, mientras que la TDT mediante la técnica de multiplexación utiliza siete múltiples digitales para la transmisión de 23 canales de TV y 19 emisoras de radio que dan cobertura a nivel estatal.

3- El estándar que se encarga de definir el sistema de modulación es DVB-T.

Televisión digital por satélite

En la televisión digital vía satélite la señal de televisión se transmite por medio de satélites que se encuentran fuera de la atmósfera, motivo por el cual las ondas necesitan dos tramos para la transmisión: el enlace ascendente, mediante el que se produce el envío de información desde el centro de emisor al satélite y el enlace descendente que transmite esta información desde el satélite de comunicaciones hasta la zona que el satélite "ilumina" en la superficie terrestre. Para evitar posibles interferencias entre ambos enlaces, cada uno de ellos utiliza una banda de frecuencias diferente.

La principal ventaja de los sistemas de televisión por satélite es la facilidad de llegar a lugares remotos o aislados (grandes zonas de cobertura).

Televisión digital por cable

En la Televisión Digital por cable, la señal de Televisión es transmitida por medio de redes de cable (coaxial o fibra óptica).

Sobre estas redes de cable es posible intercalar junto con la señal digital de TV, otros servicios como la radio, telefonía fija y acceso a internet.

El estándar utilizado para la Televisión Digital por Cable es DVB-C, que define como sistema de modulación QAM (modulación por amplitud en cuadratura) también utilizado por la TDT.

Televisión digital por Protocolo de Internet (IPTV)

En la Televisión Digital por IP la señal se distribuye por medio de protocolos de red al usuario final, normalmente a través de conexiones de datos de alta velocidad (ADSL) que facilitan las operadoras de Telefonía como otro servicio más. Esta técnica de difusión es una de las apuestas de la empresa para la distribución de video bajo demanda (VoD), permitiendo al usuario acceder al contenido digital en cualquier momento. Este apartado se va a desarrolla en profundidad en el siguiente apartado.

Televisión digital en movilidad

Se denomina Televisión digital en movilidad al "servicio de difusión de televisión que utiliza la tecnología digital para distribuir la señal y que pueda ser recibida por dispositivos móviles o portátiles (teléfono móvil, ordenador portátil, PDA, etc.)" [0].

En este entorno las operadoras de servicios facilitan dos tipos de difusión:

1- **Unicast**-. se establece un canal de comunicación exclusivo entre el emisor y el dispositivo móvil. Este método es el que utiliza la empresa, para cada uno de los dispositivos que conecta el usuario para reproducir un canal.

2- **Broadcast**-. En este servicio de difusión el emisor distribuye la señal para que pueda ser recibida por cualquier dispositivo móvil que se conecte a la red, sin limitación del número de usuarios.

IPTV

Según lo establecido por la Unión Internacional de Telecomunicaciones (ITU) en su documento ITU-T FG IPTV [0] se define la televisión por protocolo de internet (IPTV) como "servicios multimedia tales como televisión, audio, texto, gráficos, datos transmitidos sobre una red IP gestionada para entregar los niveles requeridos de calidad y experiencia de servicio, así como también seguridad, interactividad y confiablidad".

No obstante, de manera más sencilla se puede definir IPTV como un "sistema de distribución de contenido de televisión y video que, en vez de ser entregado de forma tradicional, es el espectador quien accede bajo demanda, utilizando para ello las tecnologías de acceso a Internet que ofrecen los proveedores de servicios".

Cualquier persona que tenga una conexión a internet puede ver televisión utilizando el ancho de banda facilitado por una operadora de servicios, pero ese ancho de banda es compartido con el resto de navegación web y otras utilidades del PC, sin embargo, lo fundamental de IPTV es que mediante suscripción la operadora facilita una red privada segura, con un ancho de banda exclusivo y específico para la TV, lo que garantiza una mayor calidad y una mayor oferta de canales.

Características

IPTV es un protocolo que ha sido desarrollado basándose en Video-Streaming sobre el protocolo IP. Este servicio de conexión es suministrado por las operadoras (bajo suscripción) y utilizan redes privadas seguras y el ancho de banda separado en dos: Acceso a datos de Internet y Acceso a IPTV (reservado y no accesible a otros usuarios). A esta segunda parte reservada, tienen acceso los dispositivos como los ordenadores o Set Top Box que quieren utilizar IPTV para visualizar contenidos como: VoD y televisión en directo. Todo el contenido que se visualiza mediante esta tecnología, utiliza QoS (Quality of Service) para garantizar muchos de los contenidos expuestos sean válidos para todo servicio [0].

Arquitectura básica IPTV

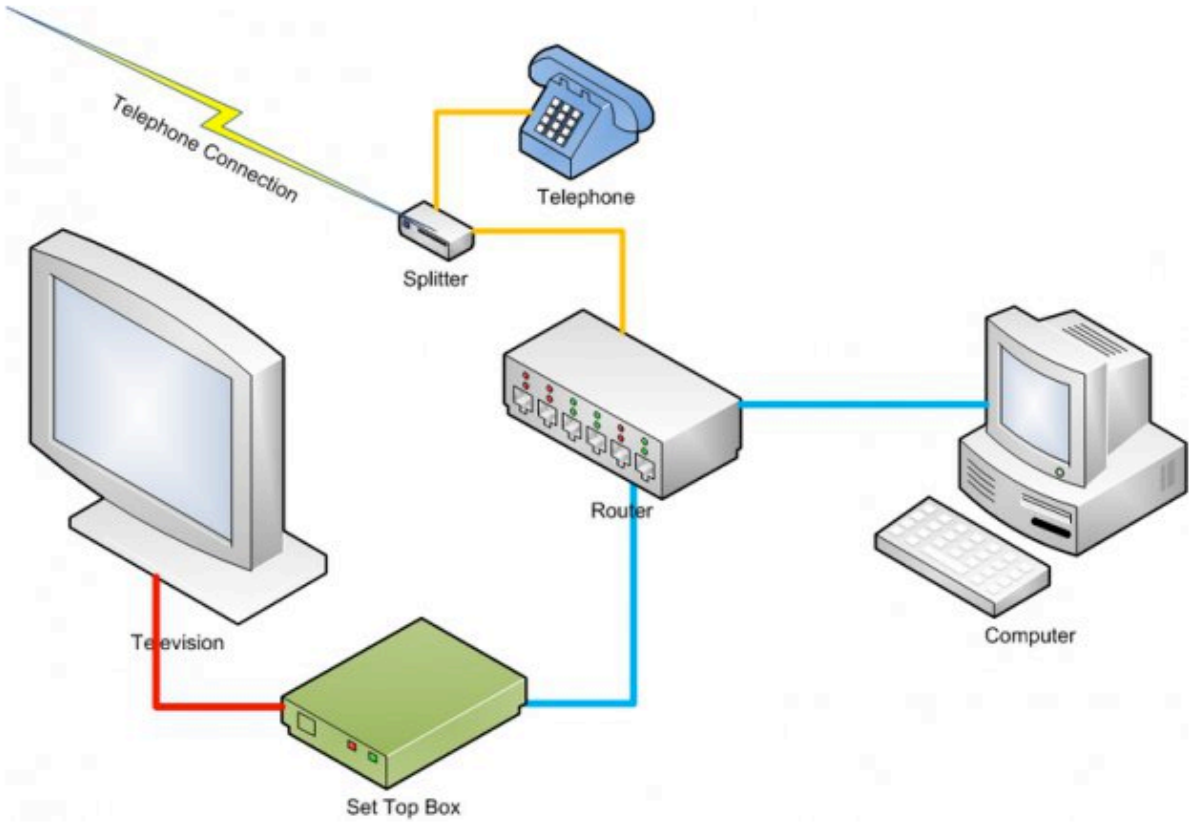


Figura 2: Ejemplo de una arquitectura IPTV [0]

En la figura 2 podemos observar una arquitectura a alto nivel de la estructura de IPTV. A través del ancho de banda que llega a los hogares (Telephone Connection), recibimos la señal de teléfono, la señal para conexiones que utilizan IP y el IPTV. Como se puede ver en la parte central de la figura 2, mediante un dispositivo Splitter, se divide la señal del teléfono, que se redirige a un teléfono fijo y la señal de internet que se redirige al Router principal. El router será el encargado de diferenciar entre IP e IPTV para que los dispositivos (Set Top Box) sean capaces de decodificar la señal IPTV y visualizar el contenido.

Requisitos del sistema:

- Conexión a Internet
- Suscripción (proveedor/distribuidora)
- Equipo de acceso:
  - Set Top Box (decodificador):

Se ha utilizado el manual que me ha facilitado la empresa para la configuración del STB a medida.

- Software capaz de decodificar señal IPTV

Tecnologías y herramientas

HTML 5:

Las siglas con que se conoce HTML corresponden a la expresión inglesa HyperText Markup Language, es un lenguaje marcado para la elaboración de Páginas Web [0]. Este estándar, fue creado por Tim Berners-Lee World Wide y Robert Cailliau y fue publicado en 1991 bajo el nombre "HTML Tags" como un nuevo sistema de "hipertexto" para compartir archivos. La versión 5 de HTML comenzó a desarrollarse en 2007 y no fue hasta el año 2010, cuando la paginas web empezaron a utilizar este estándar.

Los motivos principales por el cual he decidido realizar el desarrollo de la Página Web son:

1. Facilidad de despliegue
2. Es un lenguaje intuitivo y fácil de aprender
3. La compatibilidad con la mayoría de los navegadores

Durante la realización de la página web me he encontrado con diferentes problemas, entre los cuales tenemos:

1. La realización del diseño ha sido muy lenta, por lo que he tenido que utilizar Bootstrap para agilizar el desarrollo. Además, he utilizado una plantilla llamada "AdminLTE-3.0.0-alpha.2" para el diseño del menú principal.
2. Algunos elementos no se interpretan de la misma manera en todos los navegadores, y el diseño se ha visto afectado en algunas ocasiones.

La funcionalidad de este lenguaje es muy básica y para la realización de ciertas tareas ha sido necesaria la utilización del lenguaje Javascript, que se explica a continuación.

JAVA SCRIPT:

Javascript es un lenguaje de programación interpretado, que permite realizar páginas web dinámicas, mejorando las interfaces de usuario [0]. El código fuente es procesado por el navegador web en el lado del cliente y no en el lado servidor, permitiendo así cargar de manera rápida la página web y después ejecutar el código Javascript.

Este lenguaje se puede introducir dentro del HTML para añadir funcionalidades que HTML5 no es capaz de realizar, y es el principal motivo por el cual se ha utilizado en el desarrollo. Durante el desarrollo ha sido utilizado para añadir efectos visuales y la validación de los diferentes formularios. Además, ha permitido realizar las peticiones a la API, para tener acceso a la información almacenada en la base de datos.

Uno de los problemas que me he encontrado al realizar las peticiones mediante Javascript ha sido la seguridad, debido a que el código es visible en el *FrontEnd* y permite al usuario visualizar las peticiones que se realizan a la API para obtener o introducir información en la base de datos.

JAVA SPARK:

Java Spark es un *framework*, conjunto de biblioteca, utilizado para la realización de APIs (*Application Programming Interfaz*), permite la comunicación entre componentes software [0]. La API desarrollada, ha permitido abstraer a los diferentes componentes del sistema de la empresa de los diferentes lenguajes en los que se han realizado los desarrollos (SQL para la base de datos y HTML para la página web).

Este *framework* es muy fácil e intuitivo de utilizar, permitiendo crear un desarrollo web en Java y la posibilidad de convertirlo en una API de manera sencilla. Sin embargo, tanto la búsqueda de documentación y la configuración del mismo ha sido difícil, debido a la poca información que se puede encontrar en la web y la comunidad de usuarios tan pequeña que tiene. Únicamente, he encontrado información útil en su página web.

PHPMYADMIN:

Se trata de una herramienta de código abierto escrita en lenguaje PHP, destinada a la administración de bases de datos en SQL vía web y está disponible en setenta y dos idiomas, actualmente tiene capacidad para lo siguiente:

1. Crear y eliminar base de datos.
2. Crear, eliminar y alterar tablas.
3. Borrar, editar y añadir campos, además puede administrar los campos mediante claves.
4. Ejecutar cualquier sentencia SQL.
5. Administrar privilegios.
6. Exportar datos en varios formatos

Esta herramienta de administración de bases de datos permite facilidades de gestión mediante una interfaz gráfica sobre web muy intuitiva.

Otra de las ventajas por las que se ha escogido esta herramienta es la gran facilidad que permite en la administración de múltiples servidores.

SQL:

SQL es un acrónimo que corresponde a la expresión inglesa *Structured Query Language*. SQL es un lenguaje estándar ANSI/ISO de definición, manipulación y control de bases de datos relacionales [0]. Es un lenguaje declarativo, sólo hay que indicar qué se quiere hacer. En cambio, en los lenguajes procedimentales es necesario especificar cómo hay que hacer cualquier acción sobre la base de datos.

SQL es un lenguaje muy parecido al lenguaje natural y es muy expresivo. Por estas razones, es un lenguaje Universal que está implementado en todos los Motores de Bases de Datos, razón por la cual SQL es el lenguaje estándar de comunicación entre los diferentes Motores existentes. Esto permite acceder a todos los sistemas relacionales comerciales.

Gracias a la utilización del álgebra y de cálculos relacionales, SQL brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla.

SQL es un lenguaje declarativo de alto nivel ya que, al manejar conjuntos de registros y no registros individuales, ofrece una elevada productividad en la codificación y en la orientación a objetos. Una sentencia de SQL puede resultar equivalente a más de un programa que emplee un lenguaje de bajo nivel.

Legislación y Normativa Aplicable

La empresa Britel S.A. en el desarrollo de su actividad laboral ordinaria, se ve en la necesidad de utilizar y realizar un tratamiento de los datos de carácter personal de sus empleados y clientes.

La seguridad y privacidad de los datos de carácter personal, en concreto todos aquellos susceptibles de tratamiento informático, es un derecho fundamental que viene recogido en los diferentes ordenamientos jurídicos:

- En el ámbito de la Unión Europea, la protección de las personas físicas en relación con el tratamiento de datos personales viene recogido en el artículo 8, apartado 1 de la Carta de los derechos fundamentales de la Unión Europea [0] y en el artículo 16, apartado 1 del Tratado de Funcionamiento de la Unión Europea (TFUE) [0], en los cuales se establece que "toda persona tiene derecho a la protección de los datos de carácter personal que le conciernan".

- En el Estado Español este derecho fundamental viene recogido en La C.E. artículo 18.4 "la ley limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos" [0].

Al amparo de este compromiso legal se han desarrollado una serie de leyes que recogen los principios, derechos y obligaciones que se derivan de toda esta normativa que hay que tener en cuenta para el desarrollo del proyecto. Cabe destacar entre ellas las siguientes:

- **Directiva 95/46/CE del Parlamento y del Consejo de la Unión Europea** (24 de octubre de 1995) [0] relativa a la protección de las personas físicas, en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos.

- **Ley Orgánica 15/1999 de 13 de diciembre de Protección de datos de Carácter Personal (LOPD)** [0]**:** se trata de una ley orgánica española, en la actualidad todavía en vigor, cuyo objeto es garantizar el tratamiento de los datos de carácter personal registrados en soporte físico (susceptibles de tratamiento) y de toda modalidad de uso posterior de los datos.

- **Reglamento (EU) 2016/679 General de Protección de Datos (RGPD)** de 27 de abril [0] que deroga la Directiva Europea y parte de la LOPD: se trata de una nueva normativa europea que adapta la gestión de los datos personales a los nuevos entornos digitales, entre ellos internet y apps para móviles. Esta normativa europea añade nuevos derechos a los recogidos en la actual normativa española.
  Este reglamento fue adoptado el 27 de abril de 2016 y ha entrado en vigor el 25 de mayo de 2018, pasando a ser de carácter obligatorio para todos los estados miembros, lo que obliga a una nueva adaptación de la normativa española.

Aunque entre los objetivos de este Trabajo Fin de Master no está la adaptación a esta nueva normativa ya que excedería la pretensión del autor, sin embargo, para el desarrollo de la misma se han tenido en cuenta dos de los principios básicos recogidos en la LOPD y en el RGPD.

- **Principio de calidad del dato**: en esta aplicación se van a recoger exclusivamente los datos adecuados para la finalidad de la empresa Britel, S.A., permitiendo los derechos de acceso, rectificación, oposición y cancelación.

- **Principio de seguridad**: la aplicación "Suite Britel" va a utilizar un nivel de seguridad básico, mediante la realización de copias de seguridad periódicas y de un control de acceso a la misma. En esta aplicación se ha implementado como innovación, el algoritmo de criptografía MD5 para evitar peticiones de dispositivos no deseados.

Capítulo 3 - Análisis

En este capítulo se explican los pasos previos que se han realizado antes del desarrollo de la aplicación para definir cuáles son las herramientas a utilizar, identificar los posibles fallos que pueden ocurrir y reducir los costes. A continuación, vamos a explicar los requisitos que se han especificado con Britel S.A (requisitos de entorno y funcionales) y con el análisis funcional para definir los casos de uso de la aplicación. Continuaremos con las necesidades del autor para llevar a cabo el desarrollo y concluiremos con la planificación de cada una de las tareas.

Especificación de requisitos

La especificación de requisitos en un proyecto es de relevante importancia debido a que en ellos se va a hacer una descripción completa del comportamiento del sistema que se va a desarrollar. Por lo tanto, la consecución del objetivo va a depender del análisis y tratamiento que se haga de los requisitos a especificar, en función de los actores que intervienen en el proyecto:

- Fabricante: Iosu San Martín González
- Operador: Britel S.A.
- Cliente: consumidor de los productos y servicios.

El producto final del diseño de la aplicación software implica una buena gestión de los requisitos que se establezcan, lo que significa escuchar y entender las demandas de los usuarios finales para poderlas implementar en el sistema.

Para ello la empresa Britel S.A., como garante tanto del Operador como del Cliente tiene una serie de requisitos ya definidos y por este motivo se ha dedicado una gran parte del tiempo a mantener reuniones y contacto con el responsable de la empresa.

De los acuerdos obtenidos en las reuniones se han establecido los requisitos que se detallan a continuación:

Requisitos de entorno

El entorno es el conjunto de todos los dispositivos de hardware y software que van a formar parte del sistema. A continuación, se va a especificar la descripción del entorno que componen los diferente dispositivos software y hardware:

- El Servidor *Streaming*, que se utiliza para la difusión de los canales tiene las siguientes características: el sistema operativo CentOS 6.5 64 bit, un procesador Quad-Core CPU de 64 bit, una memoria RAM con capacidad de 32 Giga Bytes y el software instalado es Wowza Streaming Engine con la versión 4.2.

- La API, la Base de Datos, y la Página Web, se va a alojar en un servidor con las siguientes características: el sistema operativo CentOS 6.5 64 bit, un procesador Dual-Core CPU de 32 bit, una memoria RAM con capacidad de 16 Giga Bytes, un disco duro con capacidad de 2 Tera Bytes utilizado expresamente para el

almacenamiento de los datos y el software instalado es Apache con la versión 2.4 y Java con la versión 1.6_45 de Oracle.

- La empresa ha desarrollado una aplicación para visualizar los contenidos que ofrece tanto en *Streaming* como en VOD, la cual se encuentra disponible para IOS y para Android. La aplicación IOS, está disponible para todas las versiones superiores a IOS 7 o superior y para la aplicación Android, está disponible para la versión Android 6.0 o superior.

- Los Set Top Box que distribuye la empresa al cliente, disponen del sistema operativo Android. La aplicación para este tipo de dispositivo está adaptada para la captación de los botones del mando del dispositivo y únicamente disponible para las versiones de Android 4 y 5.

Requisitos funcionales:

Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que este reacciona a entradas particulares y de cómo se debe de comportar en esas situaciones.

1. Conexión a Internet:

   El servidor donde se aloja la API, debe de tener acceso a Internet los 365 días del año. En caso de no tener conexión, la API se pondrá en fuera de servicio.

2. Conexión de la base de datos:

   La primera comprobación de la API es la verificación de que la conexión con la base de datos sea correcta.

   En caso de conexión errónea, la API se pondrá en fuera de servicio y enviará un correo al director del Departamento de IT.

3. API en servicio:

   La API estará en servicio si y solo si, tiene conexión a internet y tiene comunicación con la base de datos.

4. API fuera de servicio:

   La API estará fuera de servicio, en caso de ocurrir algún error inesperado que no asegure la integridad del funcionamiento.

5. Visualizar información:

   La información se visualiza mediante la página web, realizando peticiones a la API. La API devolverá la información en formato JSON.

6. Insertar información:

   La página web dispondrá de botones para introducir información y será guardada en la base de datos realizando peticiones a la API.

7. Modificar información:

   Toda la información mostrada en la página web que esta almacenada en base de datos podrá ser modificada mediante el botón editar que dispone la página web y la API actualizara dicha información en la base de datos.

8. Eliminar información:

   La información mostrada en la página web podrá ser eliminada mediante el botón eliminar. La API, borra dicha información en función del usuario y las dependencias de dicha información en la base de datos.

9. Buscar información:

   La información visualizada en la página web sobre las peticiones se muestra en tablas, y dispondrá de un campo de búsqueda para facilitar la búsqueda de información. Además, tendrá la posibilidad de mostrar un numero de filas de la tabla a elegir entre 10,25,50,100.

10. Autentificación de usuarios:

    Los usuarios que deseen acceder a la página web, deberán introducir usuario y contraseña. La información de los usuarios que pueden acceder, estarán registrados en la base de datos y en función del rol, podrán acceder a toda la página web o a parte de ella.

11. Requisitos de fiabilidad:

La fiabilidad de los sistemas va directamente relacionada con la especificación de requisitos de diseño, averías en el Hardware e interferencias transitorias o permanentes en las comunicaciones. Es por ello, que se han definido dos formas para aumentar la fiabilidad del proyecto:

12. Prevención de fallos:

Esta especificación trata de evitar que los fallos en los sistemas se introduzcan antes de ponerlo en funcionamiento. Para ello, se ha realizado una especificación del software a realizar de forma rigurosa y se han utilizado lenguajes con abstracción de datos y modularidad. Además, se han realizado pruebas sobre el sistema que se especifica en el capítulo Implementación y Pruebas.

- Tolerancia de fallos:

  Esta especificación permite al sistema actuar en tiempo real ante los errores que puedan ocurrir durante su vida útil. Para ello, en función del error que se produzca, la tolerancia del sistema al fallo será completa, parcial o parada segura.

- Tolerancia Completa:

  El sistema sigue funcionando en su totalidad al menos durante un tiempo en caso de que ocurran errores relacionados con la perdida de conexión a red momentánea o errores controlados por software que no influyan en la integridad de la base de datos.

- Tolerancia Parcial:

  El sistema sigue funcionando, pero algunas de las funcionalidades no están operativas. Estos errores pueden ocurrir debido a una previa inserción de datos errónea en la base de datos, que produzcan errores en algunas de las consultas de la API sobre la base de datos.

- Parada segura:

  El sistema se detiene completamente y está fuera de servicio, en caso de no encontrar conexión a internet o la base de datos no está funcionando. Además, el

sistema se detendrá por completo en caso de que salte una excepción no controlada durante el desarrollo de la API o la página web.

Análisis funcional

En este apartado se va a mostrar mediante diagramas, el análisis realizado del proyecto.

Diagrama de casos de uso:

La figura 3 que se muestra a continuación, representa desde una visión general los diferentes casos de uso, mediante una descripción de los diferentes comportamientos del sistema al afrontar las tareas encomendadas por los dos "actores" principales que intervienen en el sistema: cliente y administrador.

Ambos actores pueden gestionar los canales, los dispositivos, los paquetes, la EPG (Guía Electrónica de Programas) y las parrillas. Sin embargo, como se puede observar, el actor "Administrador", es el único que puede realizar el proceso de gestión de organizaciones (clientes) y de usuarios que pueden utilizar la página web.

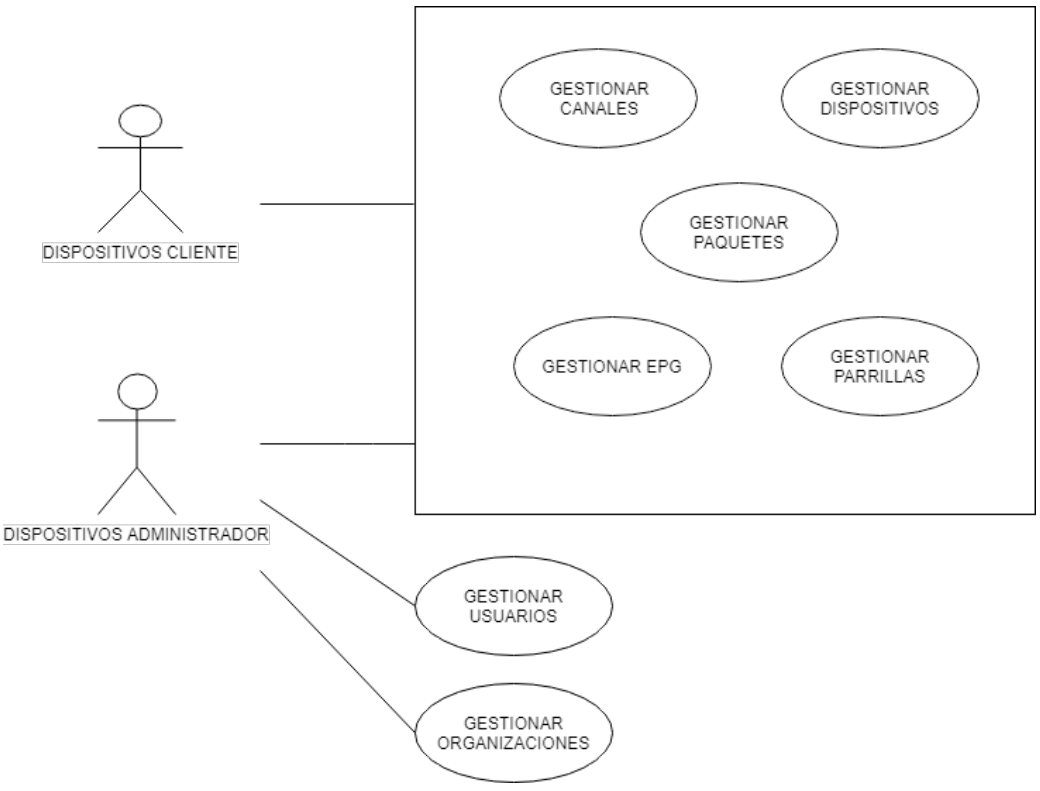


Figura 3: Casos De Uso

Casos de uso:

En este apartado se van a representar mediante diagramas, detalladamente, cada uno de los casos de uso que se han descrito en la figura 3:

- Gestionar Canales, Organizaciones y usuarios:

  Estos tres casos de uso tienen en común las siguientes opciones: crear, modificar y eliminar, como se puede observar en los tres diagramas de las figuras 4, 5 y 6.

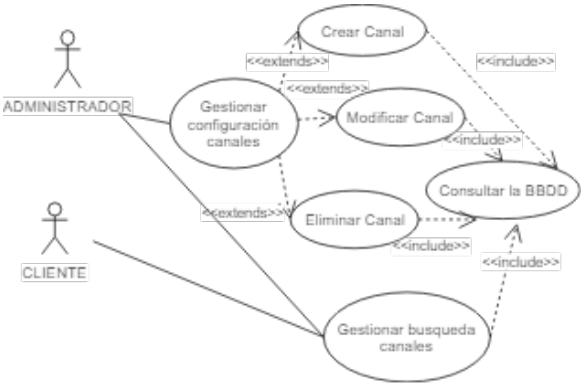


Figura 4: Caso de Uso Canales

Figura 5: Caso de Uso Organizaciones

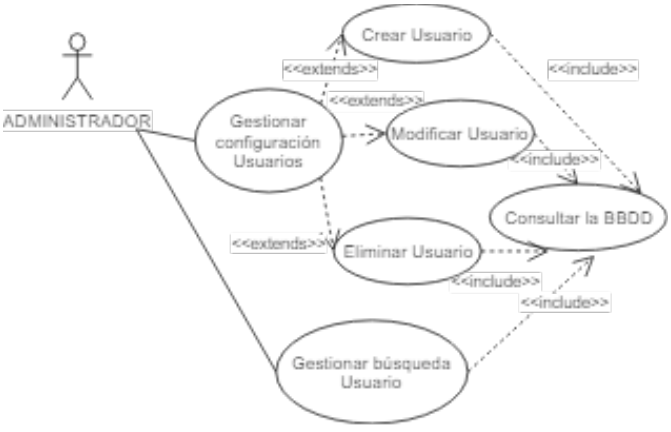


Figura 6: Caso de Uso de Usuarios

- Gestionar Paquetes:

  Los paquetes están compuestos por canales, como se observa en la figura 7 la creación implica añadir al menos un canal al paquete para que se pueda realizar este proceso. Sin embargo, la modificación incluye de manera opcional las posibilidades de creación y eliminación de un canal. Finalmente, podemos observar en la figura 7, que la eliminación de un paquete implica la eliminación de todos los canales asociados a él.

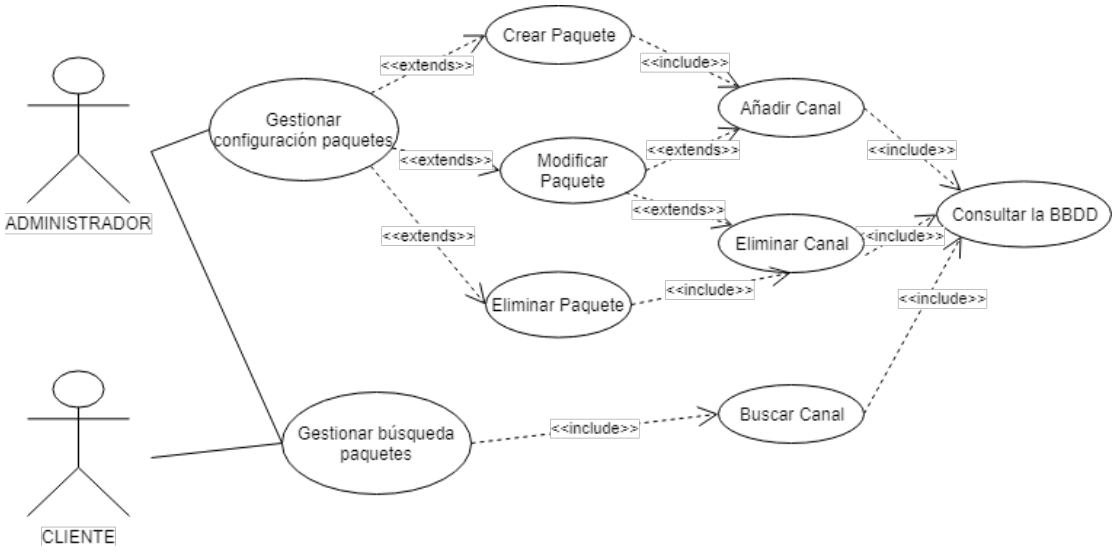


Figura 7: Casos de Uso Gestionar Paquetes

- Gestionar EPG y Dispositivos:

La EPG corresponde a un canal y los dispositivos corresponde a un único cliente. Esta relación que observamos en ambos casos, implica que, en primer lugar, siempre se deberá realizar una búsqueda para la configuración. En el caso de uso de la EPG, como se observa en la figura 8, lo primero es la búsqueda del canal correspondiente y en el caso de uso de los dispositivos, como se observa en el figura 9, lo primero es la búsqueda de la organización a la que corresponden los dispositivos a configurar. Además, como se puede observar en ambas figuras, una vez seleccionado el canal o la organización, podemos realizar los procesos: creación, modificación, eliminación y búsqueda.

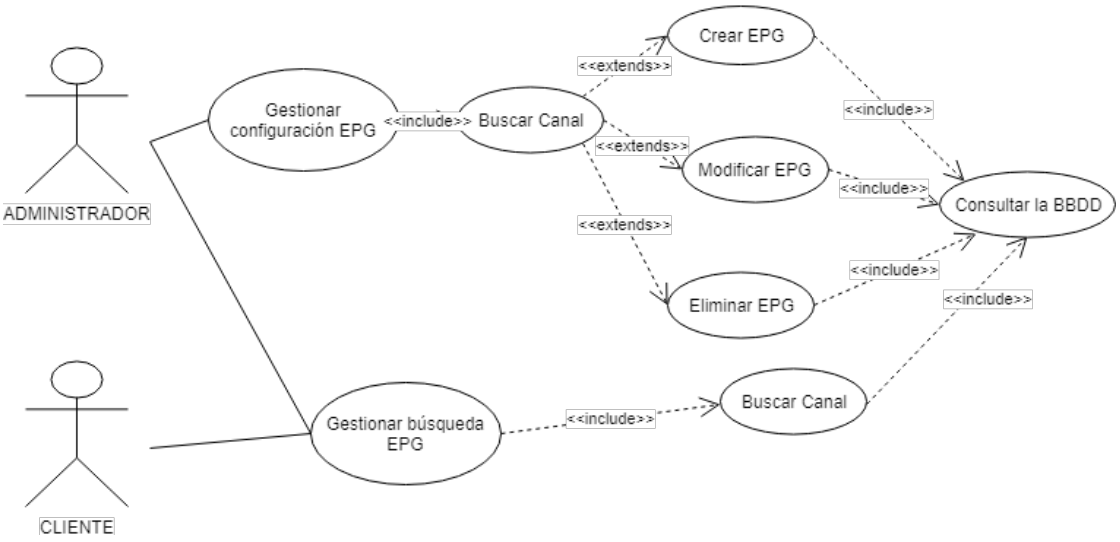


Figura 8: Casos de Uso Gestionar EPG

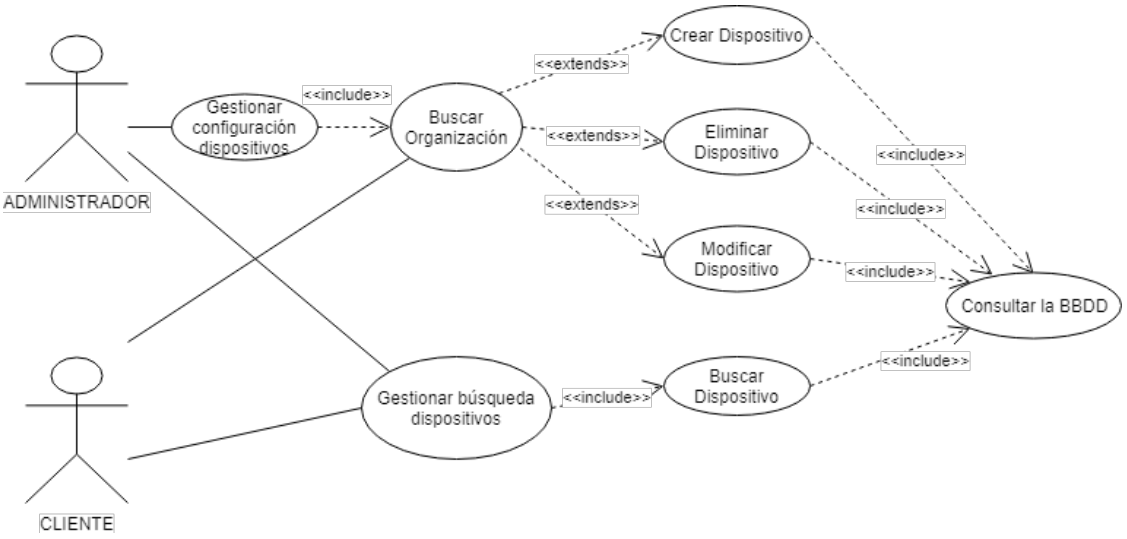


Figura 9: Casos de Uso Gestionar Dispositivos

- Gestionar Parrilla:

Una parrilla está compuesta por canales y paquetes, lo que implica la utilización de la búsqueda de canales hemos visto en las figuras 4 y 7. Respecto a la configuración de la parrilla, como podemos observar en la figura 10, para la creación de una parrilla implica de manera obligada añadir al menos un paquete o un canal. Además, también podemos observar que la eliminación de una parrilla completa, implica la eliminación de todos sus paquetes y canales asociados. Sin embargo, la modificación de una parrilla, nos da la posibilidad tanto de añadir o modificar canales o paquetes. En la parte inferior de la

figura 10, se observa una diferencia respecto a los anteriores casos de uso, ya que la búsqueda de una parrilla, implica al sistema tener que buscar los canales y paquetes asociados para poder mostrarla completamente.

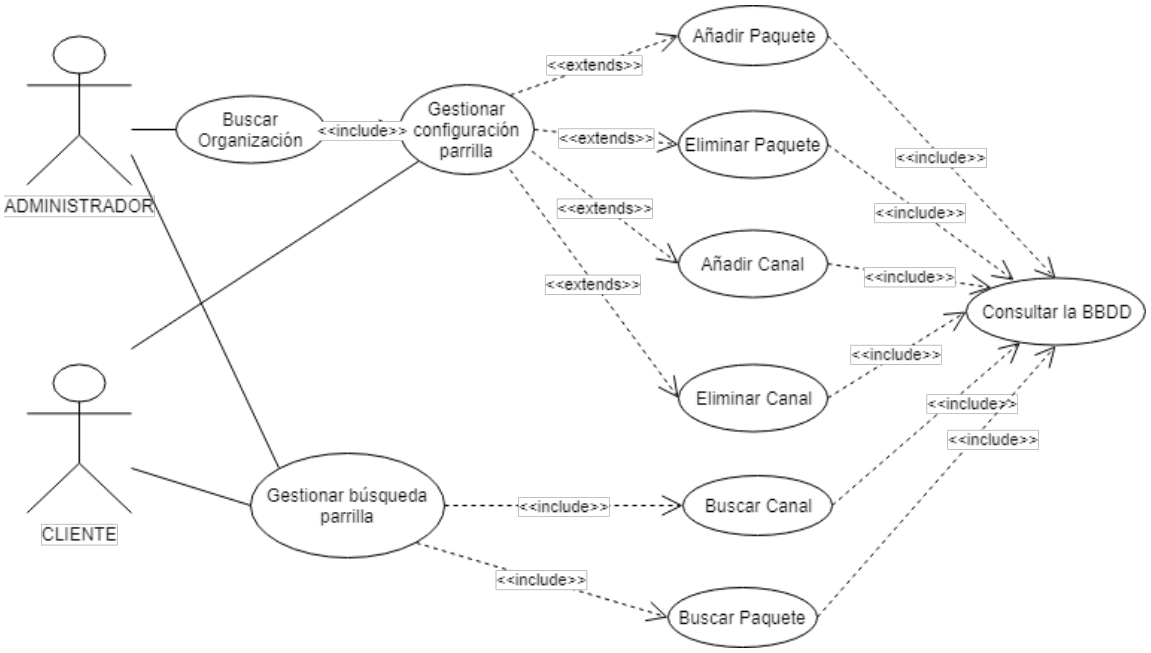


Figura 10: Casos de Uso Gestionar Parrilla

Según se observa en los diagramas, en las figuras 5 y 6 únicamente aparece el "Administrador", pues el "Cliente" no puede realizar ninguno de esos procesos.

Como se puede observar en las figuras 4, 7 y 8, el actor "Administrador" puede realizar tanto búsquedas como configuraciones mientras que el actor "Cliente", únicamente puede realizar búsquedas.

Sin embargo, como se observa en las figuras 9 y 10, los actores "Administrador" y "Cliente" pueden realizar tanto búsquedas como configuraciones. En ambas figuras, se puede observar que el "administrador" dispone de una "búsqueda de organización" antes de realizar las configuraciones. Esta búsqueda no está disponible para el "Cliente" debido a que únicamente puede acceder a las configuraciones de la organización a la que pertenece.

Además, en todas las figuras referenciadas en este apartado, podemos observar que cada uno de los procesos implica una consulta a la base de datos para guardar los cambios realizados.

Diagrama de actividad principal:

En el diagrama de la figura 11, se pueden observar los diferentes estados por los que pasa la API desde su inicio hasta la puesta en funcionamiento.

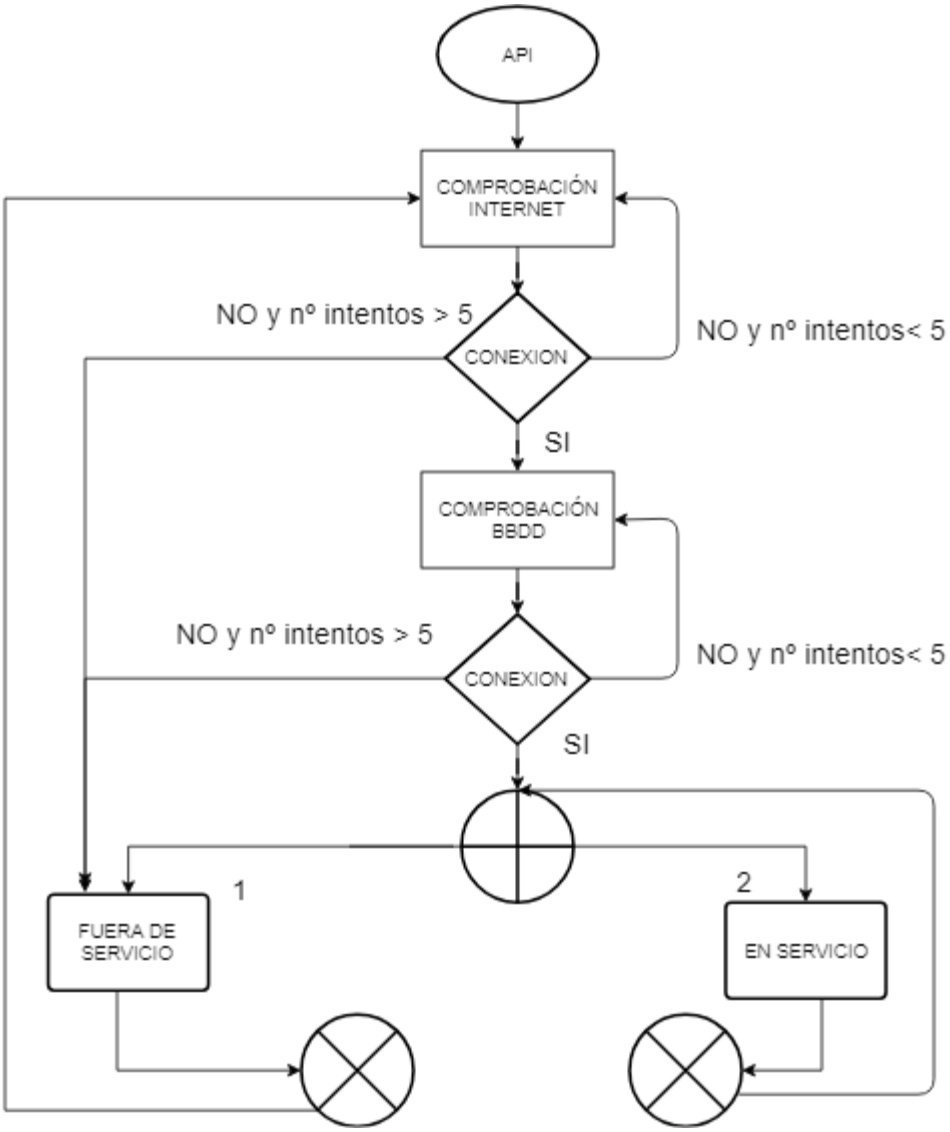


Figura 11: Actividad principal

Inicialmente, en la parte superior de la figura 11, la API realiza dos comprobaciones: la comprobación de la conexión a internet y la comprobación de la comunicación con la base de datos. El resultado de ambas comprobaciones, indicaran si pasa al estado “en servicio” o “fuera

de servicio". Estos dos estados se pueden observar mediante los números 1 y 2 en la parte inferior de la figura.

Estas dos comprobaciones, tendrán un máximo de cinco intentos que, en caso de no obtener respuesta positiva, la API pasará al estado 1 ("Fuera de Servicio") impidiendo peticiones por parte de los clientes. Cuando se encuentra en este estado, se vuelven a realizar las comprobaciones, cada diez minutos, para intentar inicializar la API en el estado 2 "en servicio".

En caso de que las comprobaciones sean positivas, la API pasará al estado 2 ("En Servicio"). Cuando se encuentra en este estado, pueden ocurrir errores por fallos en la aplicación desarrollada, por lo que, si la aplicación no es capaz de solventar el error, la API debería pasar al estado 1 "Fuera de Servicio".

Necesidades

En este apartado se van a reflejar los requisitos necesarios del autor para poder afrontar el proyecto:

1. Puesto de trabajo con un ordenador personal.

2. Acceso al CPD (Centro de Protección de Datos), donde se encuentra el servidor de Streaming.

3. Protocolos de comunicación utilizados para la API.

4. Manuales de instalación y de usuario del STB.

Planificación

Mediante el diagrama de Gantt, que se observa en la figura 12, se muestra el tiempo de dedicación para las diferentes tareas o actividades realizadas a lo largo del proyecto.

El eje superior, indica el número de semanas, mientras que el eje lateral izquierdo indica la tarea a realizar. El objetivo de este trabajo se ha establecido durante la primera semana y la especificación de requisitos durante las ocho siguientes semanas, realizando al mismo el análisis de la situación de la empresa que ha durado tres semanas. Para la realización del diseño de las aplicaciones, se ha invertido cuatro semanas y para su desarrollo e implementación diez semanas. Finalmente, podemos observar en la figura, que se han realizado la implementación y

las pruebas sobre la estructura que tiene la empresa durante dos semanas. La realización de la memoria se ha ido realizando desde que comenzó el desarrollo de la aplicación y se ha finalizado una semana después de la implementación del proyecto en la empresa.

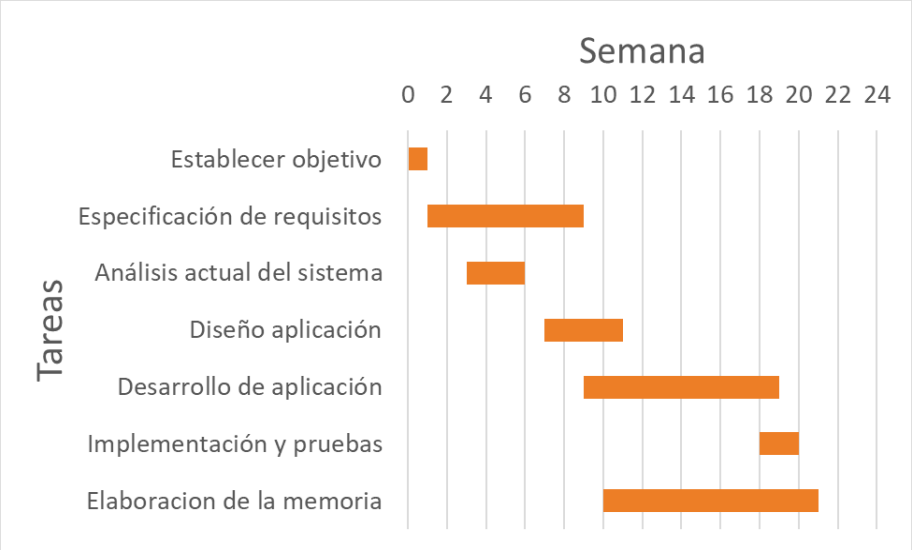


Figura 12: Resumen de Tareas

Capítulo 4 - Desarrollo

En este capítulo explican los diferentes desarrollos realizados para llevar a cabo el trabajo. Comenzaremos con el desarrollo realizado de la aplicación software principal API, continuaremos con el desarrollo de una página web básica, para poder visualizar su funcionamiento, y la explicación de la estructura de la base de datos diseñada. Finalizaremos con los protocolos de comunicación utilizados entre la API y los dispositivos y la referencia al repositorio utilizado, donde se encuentra el código libre para su visualización y utilización.

API

En esta subsección, vamos a explicar la configuración del entorno a desarrollar, continuaremos con un diagrama de clases, para observar la estructura de la aplicación, y la explicación de las funciones más relevantes que se han implementado. Finalizaremos con un diagrama de actividad para explicar cómo actúa la API al recibir una petición.

Configuración del entorno de desarrollo

El entorno de desarrollo utilizado para la API ha sido Eclipse Neon.3 Release (4.6.3), que se puede descargar desde su página web oficial [25] y está disponible para diferentes sistemas operativos como Windows, Mac y Linux. En primer lugar, crearemos un proyecto de tipo Maven y añadiremos las dependencias necesarias en el archivo POM que se genera. Finalmente, realizaremos la conexión con la Base de Datos con la herramienta Hibernate, para manipular la información.

La configuración que se ha realizado, se explica de forma más detallada en el más adelante

Diagrama de clases

Una vez tenemos el entorno de desarrollo configurado, pasamos a estructurar en paquetes y clases las diferentes funcionalidades. Mediante el diagrama de la figura 14, se describe la estructura estática de la API mostrando las clases del sistema y las relaciones entre ellas.

Las relaciones del diagrama son todas uno a uno, debido a que cada una de las peticiones va a generar un controlador y un servicio. En caso de que llegue otra petición, se creará otro controlador diferente con su correspondiente servicio. Como se puede observar, para cada una de las funcionalidades principales, tenemos un controlador que se encargará de atender la petición y devolver al cliente en formato JSON el resultado.

En la figura 13, se observan los diferentes paquetes que se han creado para organizar las clases. Adicionalmente a los paquetes que contienen los controladores y servicios, hemos creado un paquete, donde se generarán los modelos que se han definido en la Base de Datos (Model), otro con las clases que se encargan de transformar estos modelos a JSON (modelToJson), otro con la clase que se encargan de la conexión con la Base de Datos y por último un paquete que contiene funcionalidades de ayuda para que utilicen los Servicios.

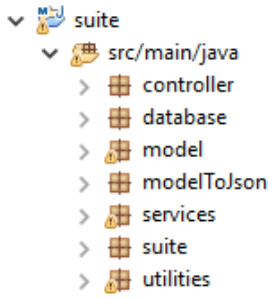


Figura 13: Vista de paquetes

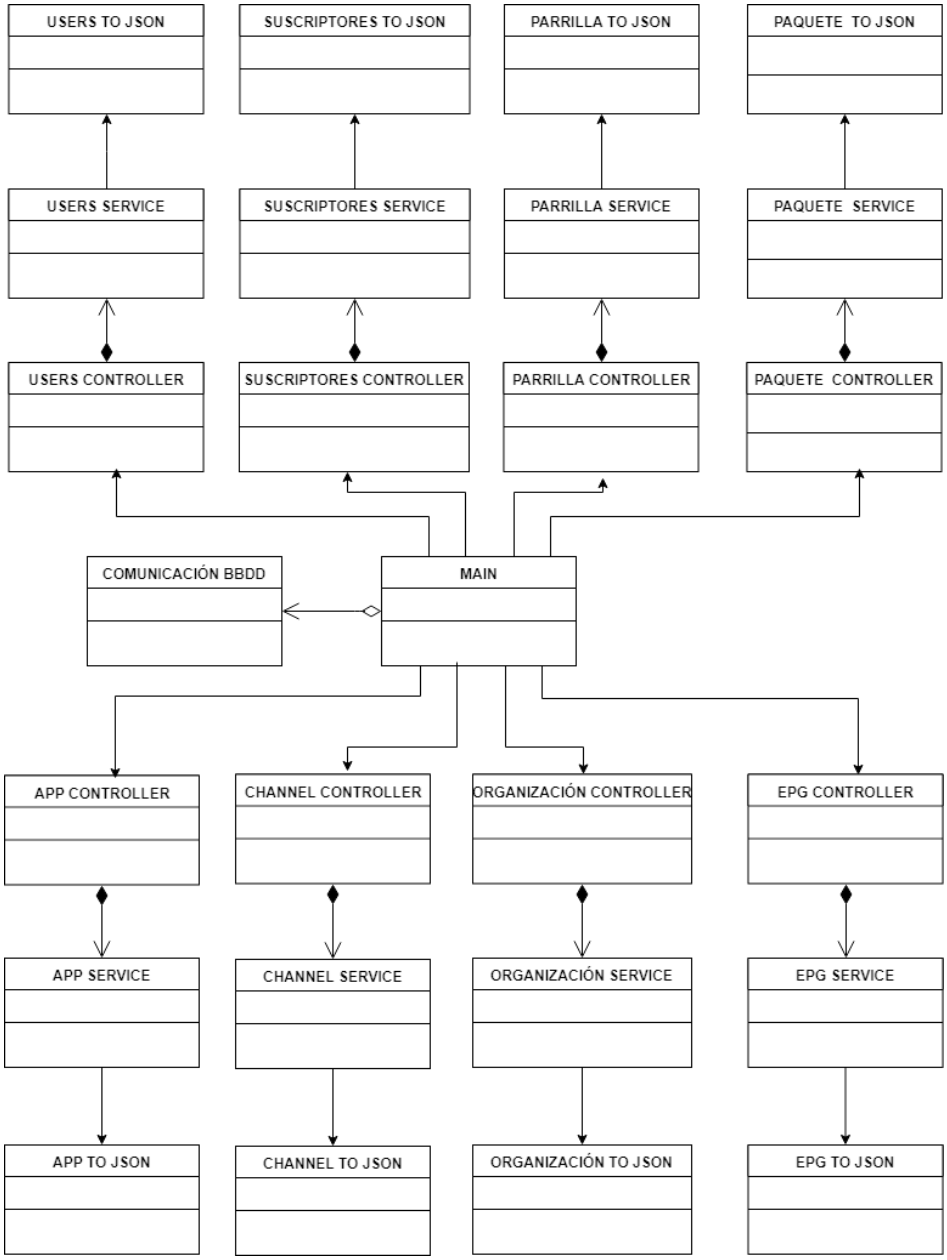


Figura 14: Diagrama de Clases

Funciones

A continuación, se reflejan algunas de las funciones más importantes que realiza la aplicación:

1. Conexión Base de Datos:

La API establece la conexión con la base de datos mediante los parámetros definidos en la herramienta que hemos configurado anteriormente, Hibernate.

Como se muestra en la figura 15, hemos definido dos funciones: la primera para iniciar la conexión y la otra para realizar "commit" y terminar la conexión. Estas dos funciones las instancia el controlador y es utilizada por el servicio, cada vez que quiere modificar un modelo de datos.

```
public class Conexion {

    private Session sesion;

    public Session iniciaOperacion()
    {
        SessionFactory sessionFactory = new Configuration().configure().buildSessionFactory();
        sesion = sessionFactory.openSession();
        sesion.getTransaction().begin();
        return sesion;
    }

    public void terminaOperacion()
    {
        sesion.getTransaction().commit();
        sesion.close();
    }
}
```

Figura 15: Función de Conexión

2. Gestionar peticiones:

   La clase principal "Main", que se muestra en la figura 16, es la encargada de instanciar todos los controladores disponibles, para que estén escuchando en el puerto configurado y gestionen las peticiones que llegan.

```
public class Main {
    public static void main(String[] args) {
        port(4567);
        new UserController(new UserService());
        new OrganizationController(new OrganizationService());
        new ChannelController(new ChannelService());
        new SuscriptorsController(new SuscriptorService());
        new AppController(new AppService());
        new EpgController(new EpgService());
        new PaquetesController(new PaquetesService());
        new ParrillaController(new ParrillaService());
    }
}
```

Figura 16: Función Principal

Dentro de cada uno de estos controladores, se van a definir las funciones de servicio que se van a ejecutar en función del tipo de petición que llega (GET o POST) y la estructura que tiene la petición. Todos los controladores tienen una estructura similar, como se puede ver en la figura 17. En primer lugar, se instancia la conexión con la base de datos. A continuación, se definen las funciones GET, como se puede ver en la figura. El primer parámetro es la ruta y el segundo la acción a realizar en caso de llegar una petición con esta ruta. Como se puede observar en la figura, todas las acciones llaman a la correspondiente función del servicio pasando siempre como argumentos, la conexión con la base de datos y los parámetros de la petición recibida.

```
public class ChannelController {
    public ChannelController(final ChannelService channelService) {
        Conexion connection = new Conexion();
        get("/channel/all", (req, res) -> channelService.getAllChannel(connection,req));
        get("/channel/alta", (req, res) -> channelService.alta(connection,req));
        get("/channel/baja", (req, res) -> channelService.baja(connection,req));
        get("/channel/modificar", (req, res) -> channelService.modificar(connection,req));
        get("/channel/obtener", (req, res) -> channelService.obtener(connection,req));
        get("/channel/obtenerepg", (req, res) -> channelService.obtenerEpg(connection,req));
    }
}
```

Figura 17: Función Controlador

3. Función All (disponible en todos los servicios):

   La función que se puede ver en la figura 18, se encarga de devolver todos los objetos del modelo que se está solicitando. La información se obtiene realizando una consulta a la base de datos y nos devuelve la lista con todos los datos almacenados.

   La figura 18 muestra un ejemplo de la función del servicio canal.

```
try {
    Session sesion = conn.iniciaOperacion();
    List<Channels> channels = sesion.createQuery("from Channels").list();
    jsonResult  = ChannelsToJson.parseChannel(channels);
    conn.terminaOperacion();
} catch (Exception e) {
    jsonResult = JsonUtil.cursoAlternativo("ACTION_NOT_ALLOWED");
}
```

Figura 18: Función Conexión BBDD

4. Función Alta (disponible en todos los servicios):

Esta función va a ser la encargada de insertar un nuevo objeto en la base de datos. Su funcionamiento se puede observar en la figura 19. En primer lugar, se comprueba que los parámetros de la petición para la creación de la nueva instancia son correctos y posteriormente, se instancia un nuevo objeto del modelo y se guarda en la base de datos.

```
Session sesion = conn.iniciaOperacion();
//Se crea el nuevo suscriptor
Organizacion nuevo_suscriptor = new Organizacion(parri,nombre,pais,region,direccion,codigopostal,ciudad,telefono,email);
//Se guarda el suscriptor
sesion.save(nuevo_suscriptor);
jsonResult  = JsonUtil.cursoNormal();
conn.terminaOperacion();
```

Figura 19: Funcion Alta

5. Función Baja (disponible en todos los servicios):

   Esta función va a ser la encargada de eliminar un objeto que se encuentra en la base de datos. En la figura 20 se puede ver que, en primer lugar, se verifica la existencia del modelo a borrar y posteriormente se ejecuta el comando **remove**.

```
Session sesion = conn.iniciaOperacion();
List<Organizacion> organizacion = sesion.createQuery("FROM Organizacion WHERE id = '" + organizationId + "'").list();
  if(!(organizacion.size()>0)){
      jsonResult = JsonUtil.cursoAlternativo("ORGANIZATION_NOT_EXIST");
  }else{
      sesion.delete(organizacion.get(0));
      jsonResult  = JsonUtil.cursoNormal();
  }
conn.terminaOperacion();
```

Figura 20: Función Baja

6. Función Modificar (disponible en todos los servicios):

   Esta función que se observa en la figura 21, va a ser la encargada de modificar uno de los objetos de los modelos almacenados en base de datos. En primer lugar, se comprueba que los parámetros de la petición son correctos y posteriormente, se actualiza los atributos del modelo.

```
Suscriptores suscriptor = suscriptores.get(0);

//Se actualiza el estado del dispositivo
suscriptor.setActivo(activo);

//Se guarda el suscriptor
sesion.update(suscriptor);
```

Figura 21: Función Modificar

7. Función Obtener (disponible en todos los servicios):

   La función, que se observa en la figura 22, se encarga de buscar el objeto que está solicitando el dispositivo a través de la petición y devolver la información sobre el modelo en formato JSON. En primer lugar, se comprueba si existe el modelo y posteriormente se envía al solicitante.

```
Session sesion = conn.iniciaOperacion();
List<Suscriptores> suscriptores = sesion.createQuery("FROM Suscriptores WHERE idsuscriptor = '" + deviceId + "'").list();
  if(!(suscriptores.size() > 0)){
      jsonResult = JsonUtil.cursoAlternativo("DEVICE_NOT_EXIST");
  }else{
      jsonResult  = SuscriptoresToJson.parseSuscriptores(suscriptores);
  }
conn.terminaOperacion();
```

Figura 22: Función Obtener

8. Función generar PID:

   La función de la figura 23, se ha creado para garantizar la procedencia de las peticiones. Únicamente los dispositivos que realicen peticiones que, conozcan cómo se genera el PID podrán utilizar la API, en caso contrario, la API le devolverá al dispositivo un mensaje de error. Para este desarrollo, hemos elegido como PID la combinación del día y mes en el que se realiza la petición, seguido del código "Md#EtC2", definido por la empresa, y finalmente codificado en MD5.

```
//Genera el PID para la comprobacion de conexion segura entre API y STB y WEB
public static String generatePid(){
    //Se coge el dia y el mes del sistema
    Date date = new Date();

    //La clave privada
    String key = "Md#EtC2";
    String mes = "";
    String dia = "";

    //Se coge el mes
    String formato="MM";
    SimpleDateFormat dateFormat = new SimpleDateFormat(formato);
    mes = dateFormat.format(date);

    //Se coge el dia
    formato="dd";
    dateFormat = new SimpleDateFormat(formato);
    dia = dateFormat.format(date);
    String cadenaPid = dia + "-" + mes + key;

    return org.apache.commons.codec.digest.DigestUtils.md5Hex(cadenaPid);
}
```

Figura 23: Función Generar PID

9. Función ParseModel:

   Esta función de la figura 24, ha sido necesaria para poder devolver al dispositivo la información solicitada a través de una petición y se ha creado una específica para cada uno de los tipos de objetos que existen en el modelo. Para ello, se encarga de ir leyendo todos los atributos de un objeto e introducirlos en una variable de tipo JSON. Finalmente, la API devuelve la variable JSON al dispositivo.

```
public class SuscriptoresToJson {
    public static JSONObject parseSuscriptores(List<Suscriptores> listaSuscriptores){
        //Objeto Json a devolver
        JSONObject obj = new JSONObject();

        for(int i=0; i<listaSuscriptores.size();i++){

            JSONObject objaux = new JSONObject();

            Suscriptores suscriptor = listaSuscriptores.get(i);

            //Se recorren todos los atributos del canal
            objaux.put("Activo", suscriptor.getActivo());
            objaux.put("Organizacion", suscriptor.getOrganizacion().getNombre());
            objaux.put("App", suscriptor.getApp().getNombre());

            obj.put(suscriptor.getIdsuscriptor().toString(), objaux);
        }

        return obj;
    }
}
```

Figura 24: Función Convertir ObjectToJson

Diagrama de secuencia

El diagrama de secuencia que se muestra en la figura 25, representa cómo interactúa cada una de los objetos del sistema cuando se realiza la petición de los canales. Se ha utilizado únicamente un diagrama ya que se puede aplicar para cualquiera de las peticiones que se realicen a la API.

Se puede observar como el actor “Dispositivo Móvil” realiza una petición a la API, para obtener todos los canales del sistema. Esta petición, es recogida por el controlador correspondiente (**Channel Controller** en este caso). El controlador identifica el tipo de petición y llama al servicio mediante la función **getAllChannels**. El servicio se encargará de la lógica de negocio y hará una consulta a la base de datos que, en función de su respuesta, generará un

resultado JSON mediante la clase **ChannelToJson** y finalmente se devolverá al usuario la respuesta final.

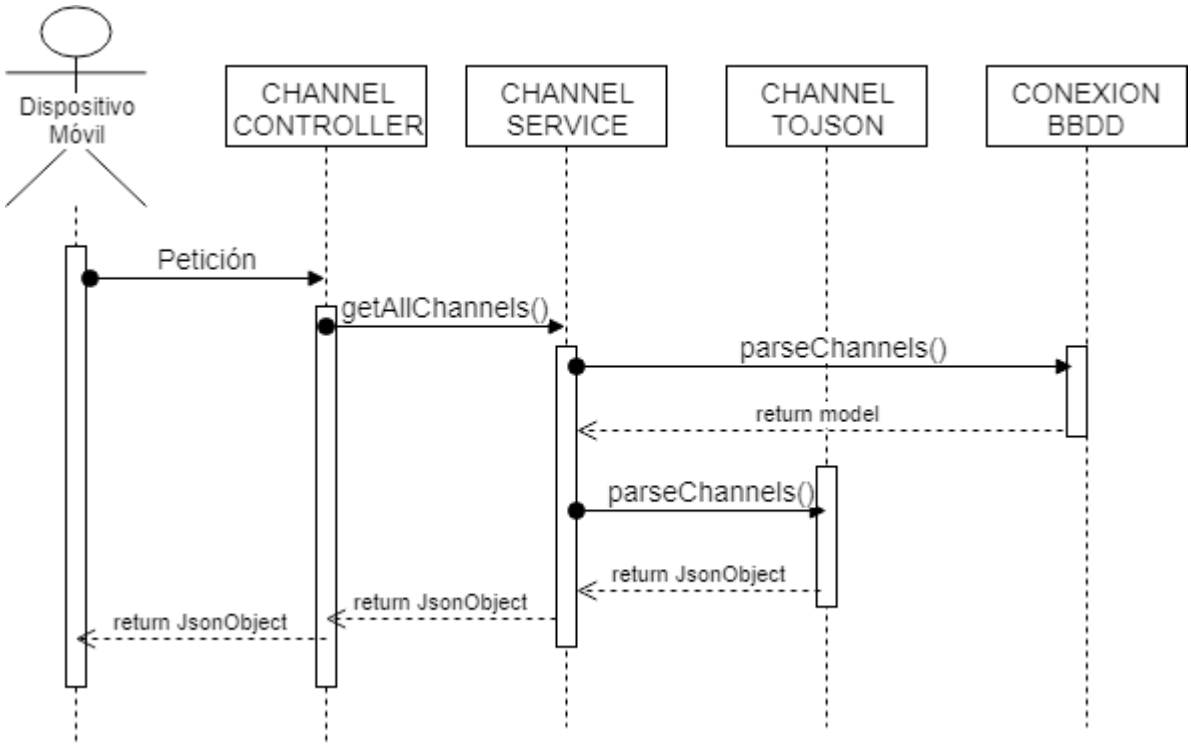


Figura 25: Diagrama de Secuencia

Página Web

En esta sección explicaremos el desarrollo de la página web encargada de interactuar con la API. Este desarrollo permite la visualización de toda la información del sistema almacenado en la base de datos. Además, la posibilidad de actualizar de dicha información, en función de las necesidades de negocio de la empresa. En primer lugar, explicaremos el diseño, continuaremos con las funciones más importantes y finalizaremos con los diferentes roles de usuarios existentes.

Diseño

El diseño de la página web se ha realizado mediante la plantilla "Admin LTE", cuyo autor es Abdullah Almsaeed y está disponible en el siguiente enlace: https://adminlte.io/.

Esta plantilla contiene las necesidades básicas de diseño como un menú en la parte izquierda, con la posibilidad de desplegar más opciones al seleccionar un elemento, e identificar al usuario conectado, mediante el nombre de usuario y una fotografía de perfil en la parte de

arriba. En la figura 26 se puede ver el diseño del menú y en la figura 27 el despliegue al seleccionar la opción de canales.

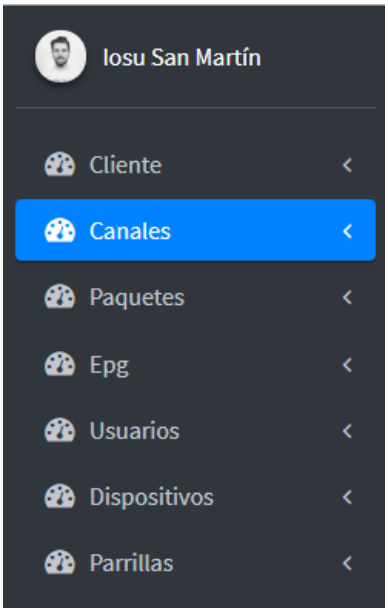


Figura 26: Menú lateral izquierdo

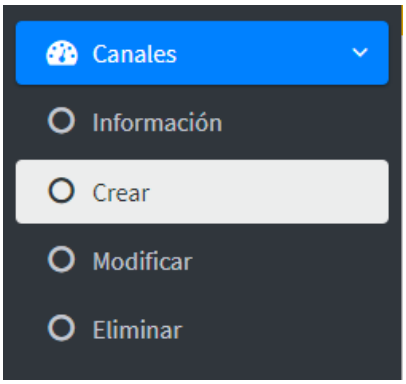


Figura 27: Despliegue Canales

Todas las páginas de la web contendrán esta plantilla para visualizar siempre el mismo menú en la parte lateral izquierda y únicamente cambiará la parte central de la página, donde se muestra el contenido en relación a la opción seleccionada. En la figura 28 podemos observar esta estructura, donde la parte que cambia sería el formulario del cliente para este ejemplo.

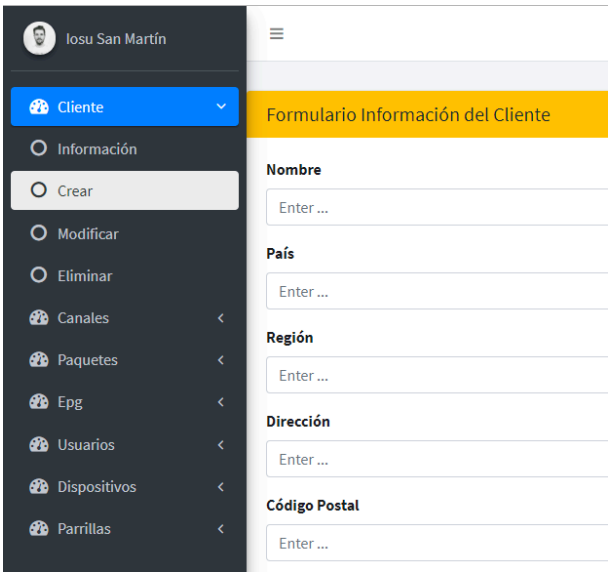


Figura 28: Estructura página completa

El proyecto se ha desarrollado con el editor de texto Notepad++. Como se observa en la figura 29, la organización de las carpetas que, contienen los archivos HTML, ha sido una carpeta para cada una de las opciones del menú izquierdo. En el interior de estas carpetas, como se ve en la figura 30, se encuentra los archivos HTML para cada una de las funcionalidades del submenú de la parte izquierda.

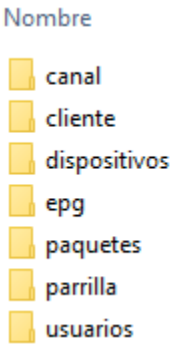


Figura 29: Organización Carpetas

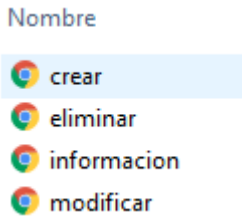


Figura 30: Archivos HTML

Funciones principales

En este apartado, explicaremos las funciones mas relevantes desarrolladas dentro de la página web para su funcionamiento:

- Petición **getAll**:

  La función que se describe en la figura 31, representa la función que se utiliza para cada una de las funciones del menú. Se puede observar que, es una petición AJAX donde el primer parámetro contiene la ruta de la petición, como segundo parámetro los argumentos de la petición y el tercer parámetro es la función a ejecutar cuando se recibe la respuesta del servidor.
  La función encargada de ejecutarse recibe como argumento el objeto JSON y como se puede ver en la figura 31, se realiza la comprobación del **Status**. En caso de ser positivo (OK), se introduce la información recibida en una tabla. En caso negativo se muestra un mensaje de error y la descripción que contiene el objeto JSON.

```
//Peticion a la api para obtener los canales
$.get( apiUrl + "/channel/all", {pid : strMD5}, function( data ) {
    $.each(data, function(i, item) {
        if(i == "status"){
            //Se elimina la tabla
            $(".firstCard").remove();
            //Se añade el status
            $(".col-12").append("<h2>" + i + ": " + item + "</h2>");
        }else if(i == "message"){
            //Se añade el status
            $(".col-12").append("<h1>" + i + ": " + item + "</h1>");
        }else{
            var tr = $('<tr>').append(
                $('<td>').text(item.LogicalChannelNumber),
                $('<td>').text(item.Nombre),
                $('<td>').text(item.Url),
                $('<td>').text(item.Description),
                $('<td>').text(item.AspectRatio),
                $('<td>').text(item.Quality),
                $('<td>').text(item.Visibility),
                $('<td>').text(item.SmallIcon),
                $('<td>').text(item.MediumIcon),
                $('<td>').text(item.MediumIcon),
                $('<td> <button type="button" class="btn btn-warning"
            );
            $("#masterBody").append(tr);
        }
    });
    $("#masterTable").DataTable();

}, "json" );
```

Figura 31: Función getAll

- Petición Obtener:

  Como se puede ver en la figura 32, le realiza una petición a la API para obtener un objeto en concreto. Para ello, esta función recibe un parámetro de identificación del objeto y se pasa como parámetro el identificado cuando se realiza la petición. De esta manera, la API sabe el identificador del objeto y hará una consulta a la base de datos. La petición AJAX que se observa en la figura, el tercer parámetro es la función que se ejecuta al obtener la respuesta. En este caso, si el resultado ha sido correcto, se llama a otra función para que rellene los datos con la información recibida.

```
function consultarInfo(id){
    //Peticion a la api para obtener la informacion del canal pulsado
    $.get( apiUrl + "/channel/obtener", {pid : strMD5, channelid : id}, function( data ) {
        $.each(data, function(i, item) {
            //Se elimina la tabla
            $("#firstCard").remove();
            if(data.status == "ko"){
                //Se añade el status
                $(".col-12").append("<h2> Message: " + data.message + "</h2>");
                //Se añade el status
                $(".col-12").append("<h1> Status: " + data.status + "</h1>");
            }else{
                $(".card-warning").show();
                rellenarInfo(item.LogicalChannelNumber,item.Nombre,item.Description,item.Ur
            }
        });
    }, "json" );
```

Figura 32: Función Obtener Objeto

- Petición Modificar:

  La figura 33, muestra la función que realiza modificaciones sobre un objeto. En primer lugar, obtiene los datos del formulario rellenado por el usuario y posteriormente, se realiza la petición a la API pasándole como argumento estos datos. El resultado que devuelve es “OK” o “KO”, que se muestra en la página web para informar al usuario del resultado de la modificación.

```
function modificarCanal() {

    var Posicion = $("#Posicion").val();
    var Nombre = $("#Nombre").val();
    var Descripcion = $("#Descripcion").val();
    var URL = $("#URL").val();
    var Aspect = $("#Aspect").val();
    var Calidad = $("#Calidad").find('option:selected').text();
    var Visibilidad = $("#Visibilidad").find('option:selected').text();
    var Pequeña = $("#Pequena").val();
    var Mediana = $("#Mediana").val();
    var Grande = $("#Grande").val();
    var id = $("#channelId").val();

    //Peticion a la api para modificar el canal
    $.get( apiUrl + "/channel/modificar", {pid : strMD5,channelid : id, logi
        if(data.status == "ok"){
            window.location = $("#linkCanalModificar").attr("href");
        }else{
            //Se elimina el formulario de modificacion
            $(".card-warning").remove();
            //Se añade el status
            $(".col-12").append("<h2> Message: " + data.message + "</h2>");
            //Se añade el status
            $(".col-12").append("<h1> Status: " + data.status + "</h1>");
        }
    }, "json" );

}
```

Figura 33: Función Actualizar

- Función PID:

  Esta función se encarga de generar el parámetro PID, requisito obligatorio para que la API atienda la petición. El PID es un parámetro compuesto por día, mes y el código “MD#EtC2” encriptado en MD5, como se puede observar en la figura 34. Además, se

puede observar en la figura la declaración de la variable que almacena la ruta de la API. En este caso la ruta es: “http://localhost:4567”

```
var apiUrl = "http://localhost:4567";

var d = new Date();
var dia = ("0" + d.getDate()).slice(-2);
var mes = ("0" + (d.getMonth() + 1)).slice(-2);
var strMD5 = $.md5(dia + "-" + mes + "Md#EtC2");
```

Figura 34: Función Generar PID

Roles de usuario

Se han desarrollado diferentes roles de usuario de acceso a la página web para restringir las funcionalidades en función del rol de usuario. A continuación, se especifica las funcionalidades disponibles que dispone cada rol:

- Rol Cliente:

  El cliente puede gestionar la información básica de su perfil, consultar todos los canales y paquetes de televisión ofrecidos por la empresa, consultar la EPG de los canales que tiene contratados, administrar los dispositivos contratados y gestionar únicamente el orden de la parrilla de canales.

- Rol trabajador Britel S.A.:

  Los trabajadores de la empresa Britel S.A. pueden realizar las mismas gestiones que los clientes, para cada uno de los clientes existente en el sistema. Además, pueden añadir nuevos dispositivos y canales a la parrilla del cliente seleccionado y modificar los canales existentes en el sistema.

- Rol Administrador:

  El administrador realiza las mismas funciones que el Rol Trabajador y puede crear o eliminar clientes, canales, paquetes, EPG, dispositivos y parrillas. Además, la gestión de usuarios de acceso a la página web se habilitará para este rol de usuario.

Base de Datos

En la figura 26, se puede observar el diseño realizado para el desarrollo de la base de datos, donde se pueden ver las relaciones entre las diferentes tablas. En primer lugar, se ha creado una tabla por cada una de las clases que hemos definido para la API: Organización, Parrilla, Canales, Paquetes, EPG, Usuarios, Aplicación, Dispositivos y VOD.

Se puede observar en la figura, que hay tablas intermedias que empiezan con la palabra *contain* como "containparripaquetes", "containcanalespaquete" y "containorganizacionusuarios". Estas tablas se han diseñado para la relación "muchos a muchos" que hay entre dos tablas, para ello se ha definido el nombre de la tabla como: la palabra "contain", seguido del nombre de una de las tablas "canales" y finalmente la otra tabla a relacionar "paquete".

Como se puede ver en la figura, los atributos de las tablas no están en su totalidad ya que se han eliminado para mostrar una imagen más clara y únicamente con la información más relevante como son sus claves primarias y foráneas.

La realización de este diseño ha pasado por varias versiones, ya que se ha puesto en conocimiento de la empresa y se ha cambiado el diseño en función de las necesidades existentes.

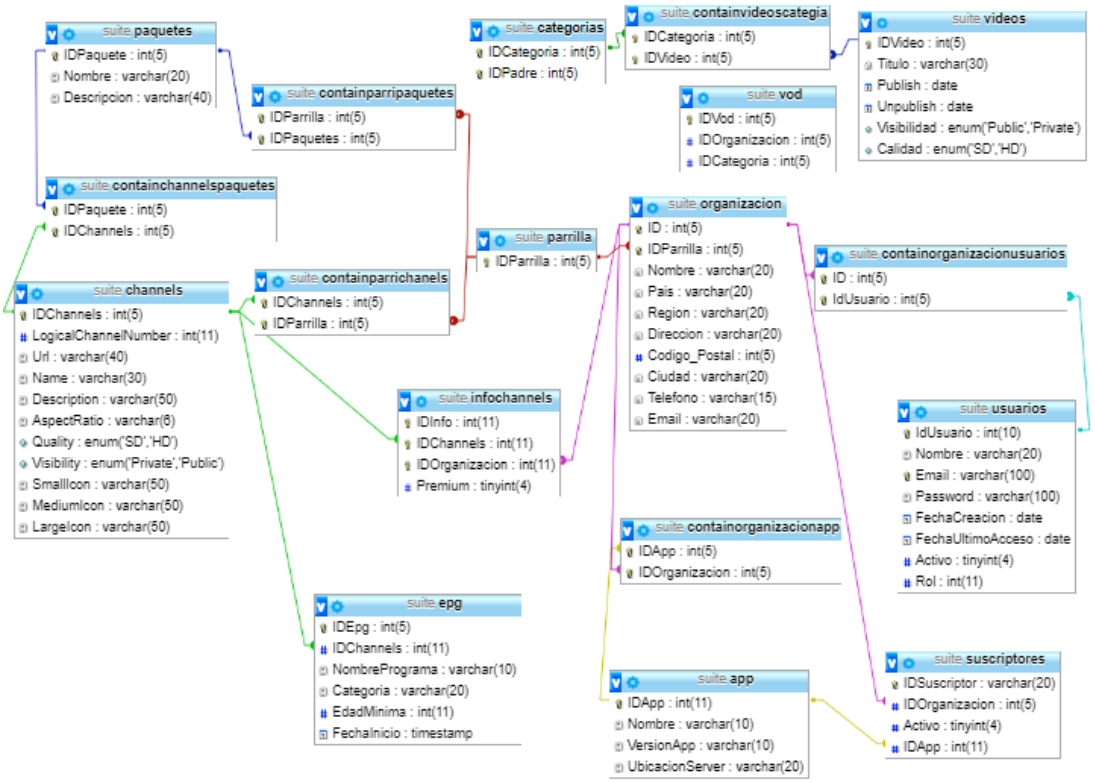


Figura 26: Diseño Base de Datos

Protocolo API-Dispositivos

La comunicación entre la API y el dispositivo se ha realizado mediante el envío de objetos JSON. Para ello, se ha establecido un estándar de comunicación que, se ha desarrollado en la parte API (nuevo) y en la parte de los dispositivos (actualización). El desarrollo para actualizar el estándar de comunicación, se ha encargado algunos de los compañeros de la empresa. De esta manera, me ha permitido realizar las pruebas entre la comunicación API y Dispositivo.

En la figura 27, se puede observar la respuesta positiva y negativa de una petición a la API que, siempre contiene un parámetro de **Status** (OK caso positivo, KO caso negativo) y otro **Message** (breve comentario sobre el resultado). Este tipo de respuestas se utilizan para las peticiones como: consultar si el dispositivo está registrado en el sistema, si el dispositivo esta activado o desactivo en el sistema, para consultar si la petición de eliminar dispositivo se ha realizado correctamente, etc. Todas estas casuísticas se pueden consultar en el Apéndice B - .

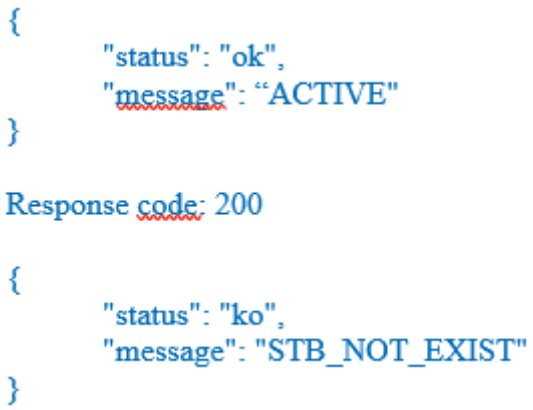


Figura 27: Respuesta Positiva o Negativa en Json

Protocolo API-Pagina Web

La comunicación entre la API y la página web se ha realizado mediante el envío de objetos JSON. Para ello se ha definido un estándar de comunicación que se describe en el Apéndice C - . En este apéndice se puede observar todas las peticiones que realiza la página y todas las respuestas posibles que puede devolver la API.

Capítulo 5 - Implementación y pruebas

Este capítulo describe los pasos que se han realizado para la implementación del proyecto dentro de la estructura de la empresa y las pruebas realizadas antes de pasar el desarrollo a producción.

Preparación previa

Antes de proceder a la realización de las pruebas es necesario establecer el sistema de acuerdo a unas premisas y condiciones, instalando el software y hardware adecuados para la implementación de la nueva API y Página Web. En primer lugar, hay que configurar un Set Top Box para actualizar la versión de la aplicación y conectarlo a través de wifi o cable Ethernet a internet. A continuación, hay que instalar el archivo *War*, que se genera mediante eclipse, para ejecutarlo en el servidor de peticiones, posteriormente se instala la base de datos diseñada, en el servidor configurado con el software Xampp y finalmente se instala la página web en un servidor, que será el encargado de realizar peticiones para mostrar la información almacenada en base de datos.

Verificación y Validación del sistema

En esta sección se explica las pruebas que se han realizado al implementar el desarrollo del proyecto.

Pruebas de sistema

La prueba de sistema consiste en arrancar la API en el servidor y verificar que el estado se encuentra “en servicio”, lo que quiere decir que la API tiene conexión a la Base de Datos y a internet. Además, se verifica el correcto funcionamiento al acceder a la página web con un usuario y contraseña y al encender un Set Top Box.

Pruebas unitarias

Las pruebas unitarias, son una forma de comprobar el correcto funcionamiento de un módulo del sistema. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. Posteriormente, con las Pruebas de Integración, se podrá asegurar el correcto funcionamiento del subsistema en cuestión.

Las comprobaciones por módulos más importantes que se han realizado han sido: gestión de clientes, canales, paquetes parrillas y dispositivos. Para cada módulo, se ha comprobado que las gestiones que se hacen desde la página web, efectivamente se realizaban sobre la base de datos.

Prueba de integración

Se realizan una vez que se han aprobado las pruebas unitarias. Únicamente se refieren a la prueba o pruebas de todos los elementos unitarios que componen un proceso.

Para este desarrollo se han realizado las pruebas unitarias para las acciones más relevantes y que pueden suponer un incorrecto funcionamiento en el sistema grave en caso de fallo. Estas pruebas son: todas las acciones que permiten eliminar, modificar parrilla de canales y añadir dispositivos a un cliente.

Resultados

Para determinar los resultados obtenidos durante la implementación y realización de cada una de las pruebas, se han tenido en cuenta los siguientes aspectos:

- Seguimiento de todo el proceso de la prueba realizada.
- Impacto provocado en la API tras la finalización de la prueba, revisando los valores de todos los campos de la base de datos y los estados en los que se encuentran los dispositivos.
- Control de mensajes de error, para aquellos casos en que los procesos no han llegado a su término, comprobando si la API ha sido capaz de solventarlos y seguir en funcionamiento.

Capítulo 6 - Conclusiones y trabajo futuro

Establecer y delimitar el objetivo de este Trabajo Fin de Master no ha sido tarea fácil, aportar alguna innovación a la empresa Britel S.A. (donde trabajo) aplicando los conocimientos adquiridos durante mis estudios de Master me ha llevado a observar y elegir entre diferentes opciones de mejora dentro de la empresa. Finalmente ha sido en el desarrollo de esta API donde he intentado aunar dos de ellas: optimizar el trabajo de los empleados y mejorar la comunicación de la empresa Britel S.A. con sus clientes.

La empresa Britel S.A. es una empresa orientada a la venta de canales de televisión, por este motivo la fundamentación teórica ha estado dirigida al estudio de la señal de video y a su difusión a través de los medios (área totalmente desconocida y que me ha motivado para su estudio). Pero es en los lenguajes de programación necesarios para desarrollar la API, donde he necesitado invertir la mayor parte de estudio.

Durante la fase de análisis y diseño de la aplicación lo más importante ha sido entender y delimitar los requisitos necesarios para el buen desarrollo de la empresa, esto ha supuesto la búsqueda y elección de herramientas y tecnologías, algunas de ellas aprendidas durante mi formación académica y otras derivadas del propio avance tecnológico, que me ha requerido un esfuerzo de aprendizaje.

En el desarrollo de la API, es donde he podido aplicar e integrar todos los conocimientos adquiridos durante mi formación, tanto académica como de mi experiencia laboral: análisis y diseño de software, gestión de bases de datos y lenguajes de programación.

En la fase de implementación del proyecto es donde he podido disfrutar comprobando los resultados obtenidos. Ha sido gratificante observar que las innovaciones e implementaciones desarrolladas en la API se han realizado con éxito, motivo por el cual esta API se encuentra instalada en la empresa Britel S.A con todo lo que supone de mejora como: la creación un cliente y su parrilla en aproximadamente treinta minutos en lugar de más de hora y media o la creación y modificación de canales en apenas unos *clicks*.

Valoración Personal

El abordar un proyecto de programación e implementación en una empresa como Britel S.A. que se encuentra en plena expansión, es un reto de gran envergadura en el que se requiere

más tiempo del que he dispuesto para llevarlo a efecto y para realizar las pruebas y el mantenimiento necesario, por lo que ha sido necesario limitar su objetivo principal.

La API ha sido incorporada a la empresa para su funcionamiento en el grado de desarrollo en la que se encuentra, pero es evidente que debido a los motivos anteriormente mencionados existen posibilidades de mejora o extensiones tanto en la API como en la página web. Respecto a la API algunas de ellas son: añadir la facturación de un cliente en función de los canales contratados, añadir la gestión de videos bajo de manada, mejorar la seguridad de conexión de los dispositivos y respecto a la página web, mejoras en el diseño y nuevas funcionalidades que se no se han llegado a completar como añadir la EPG a un canal, subir imágenes de los logos de los canales o envío de correos electrónico de la información sobre un cliente.

Capítulo 6 - Conclusions and Future Work

Establishing and delimiting the objective of this Master's thesis has not been an easy task, providing some innovation to the company Britel S.A. (where I work) applying the knowledge acquired during my Master's studies has led me to observe and choose between different improvement options within the company. Finally, it has been in the development of this API where I have tried to combine two of them: to optimize the work of the employees and to improve the communication of the company Britel S.A. with its clients.

Britel S.A. is mainly active in sale television channels, for this reason the theoretical foundation has been focused on the study of the video signal and its diffusion through the media (a totally unknown area that has motivated me to study it). But it is in the programming languages needed to develop the API that I have spent more time.

During the analysis and design phases of the application, the most important thing has been to understand and define the requirements necessary for the good development of the company, this has required the search and choice of tools and technologies, some of them learned during my academic training and others derived from my technological progress, which has required a learning effort.

In the development of the API, it is where I have been able to apply and integrate all the knowledge and skills acquired during academic training and from my work experience: software analysis and design, database management, and programming languages.

In the implementation phase of the project I have been able to enjoy seeing the results obtained. It has been gratifying to observe that the innovations and implementations developed in the API have been successfully carried out, which is why this API is installed in Britel S.A. with the improvement that it means: the creation of a client and its list of channels in approximately thirty minutes instead of more than an hour and a half or the creation and modification of channels in just a few clicks.

Personal Evaluation:

Dealing with a programming and implementation project in a company like Britel S.A., which is in full expansion, is a major challenge that requires more time than I have had to carry it out, so it has been necessary to limit its main objective.

The API has been incorporated into the company at the level of development it is at this moment, due to that reasons there are possibilities to improve or extend API and Website. Some of them are: adding a customer's billing based on the subscribed channels, adding VOD management, improving the connection security of the devices and the Website, and new features that have not been completed like add the EPG to a channel, upload images of the channel logos, and send emails of information about a customer.

Capítulo 7 - Referencias y Bibliografía

1. Gobierno de España (n.d): 4.9 Transmisión: satélite, herciana y cable. Disponible en: http://recursos.cnice.mec.es/media/television/bloque4/pag9.html [Consulta: 20 de abril]
2. El país (29 de junion de 1976): Transmisión mediante ondas hertzianas. Disponible en: https://elpais.com/diario/1976/06/29/sociedad/204847210_850215.html [Consulta: 26 de abril]
3. Wikiversidad (27 de febrero de 2016): Teoría de señales y tecnología. Disponible en: https://es.wikiversity.org/wiki/Teor%C3%ADa_de_se%C3%B1ales_y_comunicaciones [Consulta: 26 de abril]
4. Leticia Santos Armajach (21 mayo 2015): Señal de video. Disponible en: https://www.adictosaltrabajo.com/tutoriales/senal-de-video/ [Consulta: 20 de abril]
5. Studio 22 (n.d): Luminancia. Disponible en: https://www.studio-22.com/enciclopedia/luminancia.htm [Consulta: 20 de abril]
6. Studio 22 (n.d): Crominancia. Disponible en: https://www.studio-22.com/enciclopedia/crominancia.htm [Constula: 8 de mayo]
7. Universidad Internacional Valencia (n.d.): Diferencias entre señal analógica y digital. Disponible en: https://www.universidadviu.es/diferencias-senal-analogica-digital/ [Constula: 8 de mayo]
8. Wikipedia (2012): Modulación (telecomunicación). Disponible en: https://es.wikipedia.org/wiki/Modulaci%C3%B3n_(telecomunicaci%C3%B3n) [Consulta: 26 de abril]
9. DefiniciónABC (n.d.): Digitalización. Disponible en: https://www.definicionabc.com/tecnologia/digitalizacion.php [Consulta: 28 de mayo]
10. DefiniciónABC (n.d.): TDT. Disponible en: https://www.definicionabc.com/tecnologia/tdt.php [Consulta: 26 de abril]
11. Sapec (n.d.): Difusión TV. Disponible en: http://sapec.es/difusion-tv/ [Consulta: 8 de mayo]

12. Unión Internacional de Telecomunicaciones (2007): UIT-T FG IPTV. Disponible en: https://www.itu.int/md/T05-FG.IPTV-R-0038/es [Consulta: 8 de mayo]
13. Wikipedia (5 de julio de 2018): IPTV. Disponible en: https://es.wikipedia.org/wiki/IPTV [Consulta: 28 de mayo]
14. Malavida (7 de marzo de 2018): Qué es IPTV y cómo funciona. Disponible en: https://www.malavida.com/es/guias-trucos/que-es-la-television-iptv-y-como-funciona-006526#gref [Consulta: 5 de junio]
15. Christopher. Murphy y Nicklas. Persson (2010): HTML y CSS ISBN: 9788441526112 8441526117. Disponible en: Biblioteca UCM, Facultad de Informática [Consulta: 10 de junio]
16. Mark Doernhoefer (2006): JavaScript ISBN: 0163-5948. Disponible en: Biblioteca UCM, Facultad de Informática [Consulta: 8 de junio]
17. Spark (n.d.): Spark Java Documentation. Disponible en: http://sparkjava.com/documentation [Consulta: 5 de julio]
18. Michael. Kofler (2001): MySQL ISBN: 1893115577 9781893115576. Disponible en: Biblioteca UCM, Facultad de Informática [Consulta: 12 de junio]
19. Diario Oficial de las Comunidades Europeas (2000): Carta de los derechos fundamentales de la unión europea. Disponible en: http://www.europarl.europa.eu/charter/pdf/text_es.pdf [Consulta: 20 de junio]
20. Diario Oficial de la Unión Europea (2010): Tratado de Funcionamiento de la Unión Europea. Disponible en: https://www.boe.es/doue/2010/083/Z00047-00199.pdf
21. Actualización Constitución Española (1992): Los derechos y deberes fundamentales. Disponible en: http://www.congreso.es/consti/constitucion/indice/titulos/articulos.jsp?ini=18&tipo=2 [Consulta: 16 de mayo]
22. Parlamento Europeo (1995): Reglamento General de Protección de Datos. Disponible en: http://www.wipo.int/wipolex/es/details.jsp?id=13580 [Consulta: 16 de mayo]
23. Gobierno de España (1999): Reglamento General de Protección de Datos. Disponible en: https://www.boe.es/buscar/doc.php?id=BOE-A-1999-23750 [Consulta: 16 de mayo]

24. Parlamento Europeo (2016): Reglamento General de Protección de Datos. Disponible en: https://www.boe.es/doue/2016/119/L00001-00088.pdf [Consulta: 17 de mayo]
25. Eclipse Foundation (n.d.): Descarga Eclipse Neon 3. Disponible en: https://www.eclipse.org/downloads/packages/release/Neon/3 [Consulta: 29 de junio]
26. Gobierno de España (n.d.): Televisión Digital. Disponible en: http://www.televisiondigital.gob.es/TelevisionDigital/Paginas/television-digital.aspx [Consulta: 29 de mayo]
27. Escuela Universitaria de Música (n.d.): Física del sonido. Disponible en: http://www.eumus.edu.uy/eme/ensenanza//acustica/apuntes/material-viejo/fisica_r/ [Consulta: 22 de mayo]
28. Fer Esposito (26 de noviembre de 2013): Luminancia, crominancia y saturación. Disponible en: https://prezi.com/j-v_34wqqnu5/luminanciacrominanciasaturacion/ [Consulta: 16 de junio]
29. Telectrónica (2008): Señales analógicas y digitales. Disponible en: https://tuelectronica.es/senales-analogicas-y-digitales/ [Consulta: 15 de junio]
30. Blog Tecno-eco (2009): Como llega la tv a casa. Disponible en: http://guiasycursos.blogspot.com/2009/06/como-llega-la-senal-de-tv-casa.html [Consulta: 15 de junio]
31. Universidad de los Andes (n.d.): Modulación y Transmisión de datos. Disponible en: https://profesores.virtual.uniandes.edu.co/~isis1301/dokuwiki/lib/exe/fetch.php?media=recursos:06_modulacion.pdf [Consulta: 15 de junio]
32. Televisión digital modulación: http://www.um.edu.uy/_upload/_investigacion/web_investigacion_53_Memoria_5_ModulacionDigital.pdf [Consulta: 15 de junio]
33. Universidad de Montevideo (n.d): Modulación Digital. Disponible en: http://www94.etc.upm.es/tsmpeg2d.pdf [Consulta: 20 de junio]
34. Tutoriales Point (n.d.): JDBC - Sample, Example Code. Disponible en: https://www.tutorialspoint.com/jdbc/jdbc-sample-code.htm [Consulta: 20 de junio]
35. Victor Robles (2014): Modelo MVC. Disponible en: https://victorroblesweb.es/2013/11/18/tutorial-mvc-en-php-nativo/ [Consulta: 20 de junio]

36. Michael Scharhag (5 de junio de 2014): Building a simple RESTful API with Spark. Disponible en: https://www.mscharhag.com/java/building-rest-api-with-spark [Consulta: 10 de julio]
37. LibrosWeb (n.d.): Breve historia de HTML. Disponible en: http://librosweb.es/libro/xhtml/capitulo_1/breve_historia_de_html.html [Consulta: 25 de junio]

Apéndice A - Configuración del entorno de desarrollo

En primer lugar, debemos crear un proyecto, donde iremos añadiendo las diferentes clases y paquetes para estructurar todo el código. El tipo de proyecto que se va a utilizar es Maven, que es una herramienta software para gestionar y construir proyectos. La creación de este proyecto se hace seleccionado File → New Project, y aparecer una ventana como se observa en la Figura A.1.

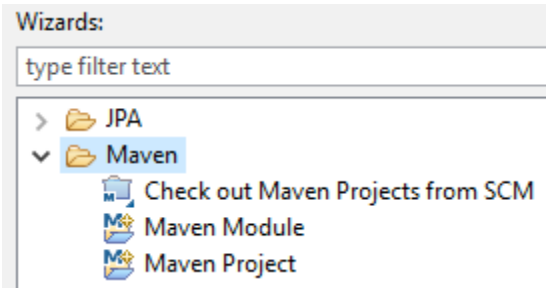


Figura A.1: Crear Proyecto

Al crear el proyecto, se genera un archivo POM (Project Object Model), que contiene información sobre la configuración y sus dependencias. Para este desarrollo se han necesitado las dependencias que se pueden observar en la Figura A.2, para la utilización de Spark (spark-core), la conexión con la base de datos (mysql-connector-java), la conversión de los objetos a JSON, la generación del modelo utilizado en la Base de Datos (hibernate-core), la configuración para actuar como servidor (jetty-server) y por último algunas dependencias para facilitar la programación en lenguaje Java.

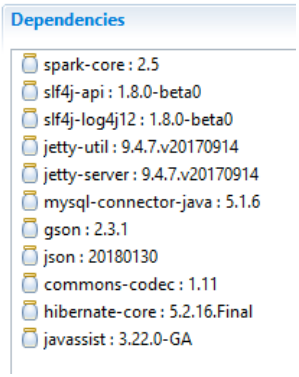


Figura A.2: Dependencias

La configuración de la conexión con la Base de Datos, se realiza mediante la herramienta Hibernate, encargada de realizar el mapeo entre una base de datos relacional y el modelo de objetos de una aplicación. Esta herramienta, no viene por defecto en la instalación del entorno Eclipse por lo que se instaló siguiendo los paso explicados en el siguiente enlace (http://www.programandoapasitos.com/2016/01/instalacion-y-configuracion-de.html).

Una vez instalada la herramienta, hay que configurarla para generar los archivos necesarios y así generar el modelo.

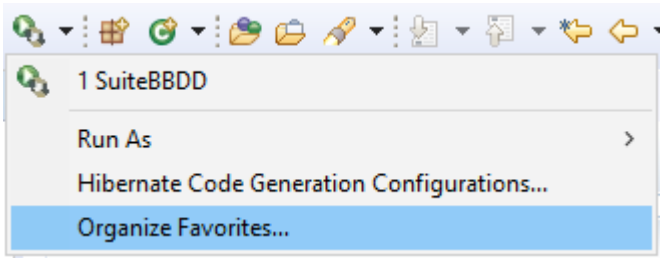


Figura A.3: Configuración Hibernate

Esta configuración se realiza mediante la opción, Hibernate Code Generation Configuraions, como se observa en la Figura A.3, y al seleccionarla, se abre una ventana, ver Figura A.4, donde configuraremos en la pestaña “Main”, la ubicación del proyecto y dónde se va a generar el modelo.

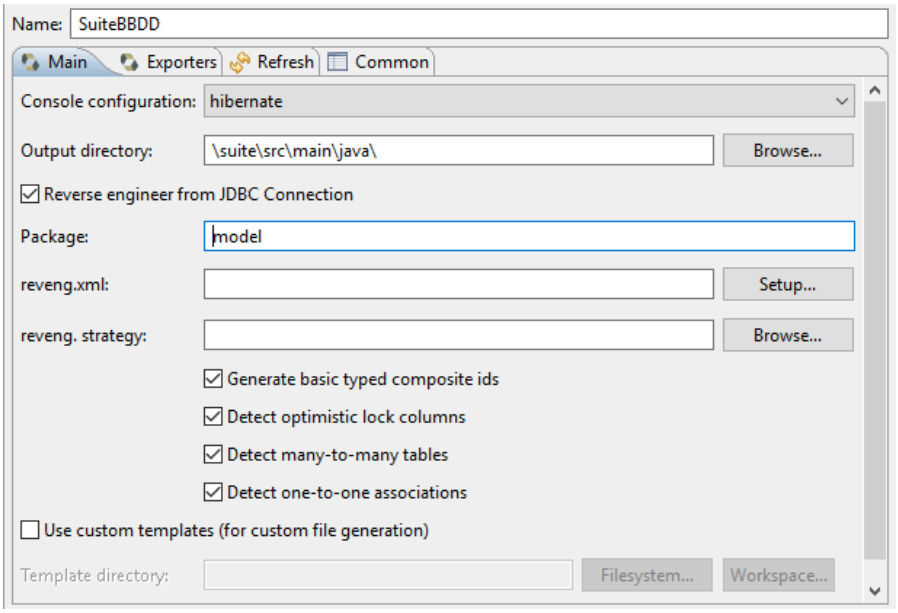


Figura A.4: Pestaña de configuración Main

En la siguiente pestaña "Exporter", ver Figura A.5, se indican los ficheros que vamos a generar de configuración. En este proyecto nos interesa el mapping que vamos a hacer con la base de datos (Hibernate XML Mappings) y la configuración de Hibernate (Hibernate XML Configuration).

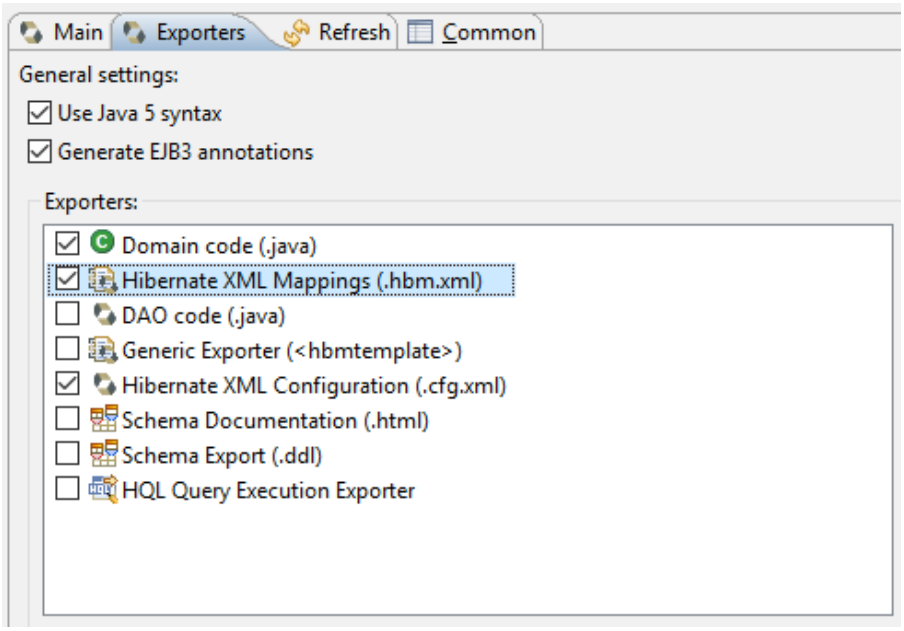


Figura A.5: Pestaña de configuración Exporter

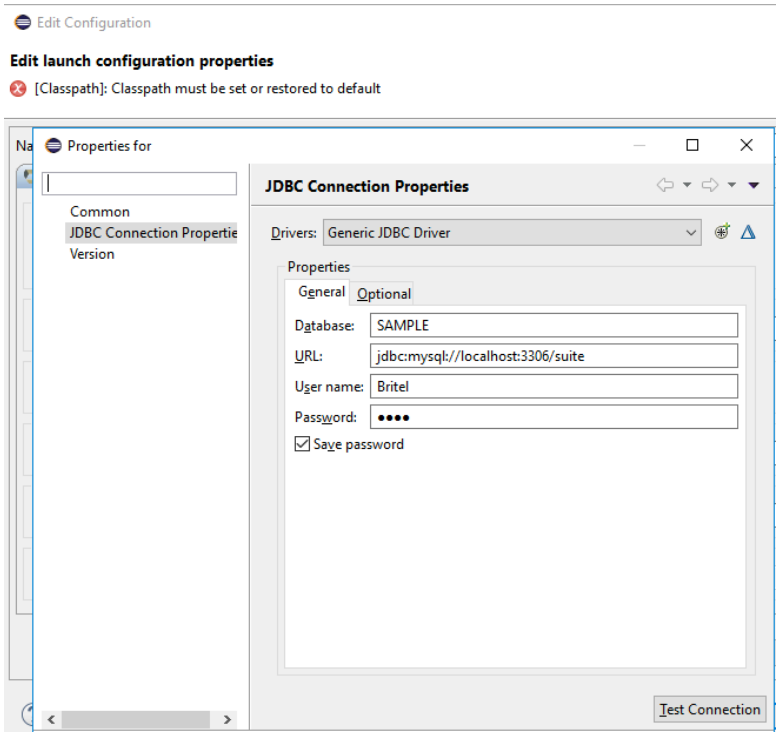
La configuración de la Base de Datos, se puede hacer de forma manual, como se puede observar en la Figura A.6, modificando el el fichero que se ha creado, al seleccionar la opción Hibernate XML Configuration en el anterior paso, o se puede realizar con la ayuda de la interfaz de Hibernate, ver Figura A.7.

hibernate.cfg.xml

```
1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2  <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "ht
3  <hibernate-configuration>
4      <session-factory>
5          <property name="hibernate.archive.autodetection"/>
6          <property name="hibernate.bytecode.use_reflection_optimizer">false</property>
7          <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
8          <property name="hibernate.connection.password">1234</property>
9          <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/suite</property>
10         <property name="hibernate.connection.username">Britel</property>
11         <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
12         <property name="hibernate.search.autoregister_listeners">true</property>
13         <property name="hibernate.validator.apply_to_ddl">false</property>
```

Figura A.6: Configuración manual

En caso de realizarlo mediante la ayuda de la interfaz de Hibernate, debemos ir a Edit Configuration → Launch Configuration → JDBC Connection Poperties y rellenar la información sobre la base de datos que se va a utilizar.



Apéndice B - Protocolo API-Dispositivos

Introducción:

Este documento especifica la versión 1 de la aplicación que gestiona la página web de Britel S.A. El servidor donde se encuentra la API y al cual se realizan las peticiones se aloja en la siguiente dirección:

URL: http://91.126.141.4

Para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:

- SHA256(dd-mmMd#EtC2):

  dd = día del mes con dos dígitos

  - (guión)

  mm = mes numérico con dos dígitos

  Md#EtC2 = clave privada

  Ejemplo:

  SHA256(22-11Md#EtC2) =
  a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa30842d5 3df3687b4

Funciones:

❖ *Registro de decodificadores:*

- Descripción:

  Dar de alta a un decodificador.

- URL:

http://91.126.141.4/stb/subscribe

- Parámetros de entrada (GET/POST)

[pid=<pid>, deviceid = <deviceid>, organziacionid = <organziacionid >, active = <active>]

- **pid**:

Para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:

- SHA256(dd-mmMd#EtC2):

dd = día del mes con dos dígitos
- (guión)
mm = mes numérico con dos dígitos
Md#EtC2 = clave privada

Ejemplo:

SHA256(22-11Md#EtC2) = a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa30842d53df3687b4

- **DeviceId:**

id de decodificador en Britel (único)

- **OrganizacionID:**

Organización a la que pertenece el STB (único)

- **Active:**

Se si queremos registrar el STB como activado (1) o desactivado (0)

- FullURL:
http://91.126.141.4/stb/subscribe?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceid=B4200044F0A0&organizacionid=123456789&active=1

- Parámetros de salida:

- Curso normal:

```
{
        "status": "ok",
    "message": OK",
}
```

Response code: 200

- Curso alternativo:

  Pid incorrecto:

  ```
  {
      "status": "ko",
      "message": "REQUEST_NOT_ALLOWED"
  }
  ```

  Response code: 400

  Id usuario existente:

  ```
  {
      "status": "ko",
      "message": "STB_ALREDY_EXIST"
  }
  ```

  Response code: 400

  Introducir algun DeviceId:

  ```
  {
      "status": "ko",
      "message": "INTRODUCE_DEVICEID"
  }
  ```

  Response code: 400

  - Introducir alguna Organizacion:

    ```
    {
        "status": "ko",
        "message": "INTRODUCE_ORGANIZACIONID"
    }
    ```

    Response code: 400

  - La organizacion no existe:

```
{
        "status": "ko",
        "message": "ORGANIZATION_NOT_EXIST"
}
```

Response code: 400

- Accion no permitida:

```
{
        "status": "ko",
        "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Eliminar Decodificadores:*

❖ Descripción

Dar de baja un usuario en el sistema.

❖ URL:

http://91.126.141.4/stb/unsubscribe

❖ Parámetros de Entrada (GET / POST):

[pid=<pid>, deviceId = <deviceId>]

- pid:
  para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:
  - SHA256(dd-mmMd#EtC2)

    dd = día del mes con dos dígitos
    - → guión
    mm = mes numérico con dos dígitos
    Md#EtC2 = clave privada

  Ej SHA256(22-11Md#EtC2) = a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa3 0842d53df3687b4

- deviceId:

id de suscriptor en Britel (único)

- FullURL:

http://91.126.141.4/stb/unsubscribe?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceId=B4200044F0A0

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": OK"
}
```

Response code: 200

- Curso alternativo
  - pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

  - id usuario inexistente:

```
{
    "status": "ko",
    "message": "STB_NOT_EXIST"
}
```

Response code: 400

  - acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

- Introducir algun DeviceId:

```
{
        "status": "ko",
        "message": "INTRODUCE_DEVICEID"
}
```

Response code: 400

❖ *Desactivar un Decodificador:*

❖ Descripción

Eliminar un usuario en el sistema.

❖ URL:

http://91.126.141.4/stb/remove

❖ Parámetros de Entrada (GET / POST):

[pid=<pid>, deviceId = <deviceId>]

- pid:
  para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:
  - SHA256(dd-mmMd#EtC2)

    dd = día del mes con dos dígitos
    - → guión
    mm = mes numérico con dos dígitos
    Md#EtC2 = clave privada

  Ej SHA256(22-11Md#EtC2) =
  a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa3 0842d53df3687b4

- deviceId:

  id de suscriptor en Britel (único)

- FullURL:

http://91.126.141.4/stb/remove?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceId=B4200044F0A0

Parámetros de Salida:

- Curso normal:

```
{
        "status": "ok",
        "message": OK"
}
```

Response code: 200

- Curso alternativo
  - pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

  - id usuario inexistente:

```
{
    "status": "ko",
    "message": "STB_NOT_EXIST"
}
```

Response code: 400

  - acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

- Introducir algun DeviceId:

```
{
        "status": "ko",
        "message": "INTRODUCE_DEVICEID"
}
```

Response code: 400

❖ *Actividad Decodificador:*

- ❖ Descripción:

  Consultar si está activo el decodificador o no en el sistema.

- ❖ URL:

  http://91.126.141.4/stb/active

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, deviceId = <deviceId>]

  - pid:
    para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:
    - SHA256(dd-mmMd#EtC2)

      dd = día del mes con dos dígitos
      - → guión
      mm = mes numérico con dos dígitos
      Md#EtC2 = clave privada

    Ej SHA256(22-11Md#EtC2) = a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa3 0842d53df3687b4

  - deviceId:

    id de suscriptor en Britel (único)

  - FullURL:

    http://91.126.141.4/stb/active?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceId=B4200044F0A0

- ❖ Parámetros de Salida:

  - Curso normal:

    {
    "status": "ok",

"message": “ACTIVE"
}

Response code: 200

{
"status": "ko",
"message": "STB_NOT_EXIST"
}

Response code: 400

- Curso alternativo
  - pid incorrecto:

{
"status": "ko",
"message": "REQUEST_NOT_ALLOWED"
}

Response code: 400

  - id usuario inexistente:

{
"status": "ko",
"message": "STB_NOT_EXIST"
}

Response code: 400

  - id usuario inactivo:

{
"status": "ko",
"message": "STB_NOT_ACTIVE"
}

Response code: 400

  - acción no permitida:

{

```
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

- Introducir algun DeviceId:


```
{
    "status": "ko",
    "message": "INTRODUCE_DEVICEID"
}
```


Response code: 400

Apéndice C - Protocolo API-PáginaWeb

Introducción

Este documento especifica la versión 1 de la aplicación que gestiona la página web de Britel S.A.

El servidor donde se encuentra la API y al cual se realizan las peticiones se aloja en la siguiente dirección:

URL: http://91.126.141.4

Para evitar peticiones externas a la función. El valor responderá a la siguiente fórmula:

- SHA256(dd-mmMd#EtC2):

dd = día del mes con dos dígitos

- (guión)

mm = mes numérico con dos dígitos

Md#EtC2 = clave privada

Ejemplo:

SHA256(22-11Md#EtC2) =

a4d49f776a7555ec631d34d076d5b93079fdb63f7a3bf1fa30842d5 3df3687b4

Funciones

Gestión de usuarios

❖ *Login de usuario*

- ❖ Descripción:

  Ingreso en la página web de un usuario.

- ❖ URL:

  http://91.126.141.4/users/login

- ❖ Parámetros de entrada (GET/POST)

  [pid=<pid>, usuario=<usuario>,pass=<pass>]

  - FullURL:
    http://91.126.141.4/users/login?pid=971b4a1e83a20fef931b6
    8bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceid=B4200044F0A0&usuario=
    <usuario>&pass=<pass>

- ❖ Parámetros de salida:

  - Curso normal:

    {

    "status": "ok",

    "message": OK",

    }

    Response code: 200

  - Curso alternativo:

    Pid incorrecto:

    {

    "status": "ko",

    "message": "REQUEST_NOT_ALLOWED"

    }

    Response code: 400

Usuario o contraseña incorrecto:


```
{
        "status": "ko",
        "message": "PASSWORD_USER_NOT_CORRECT"
}
```


Response code: 400

Usuario no activo en el sistema:


```
{
        "status": "ko",
        "message": "USER_NOT_ACTIVE"
}
```


Response code: 400

❖ *Añadir de usuario*

- Descripción:

  Alta en la página web para un usuario.

- URL:

  http://91.126.141.4/users/alta

- Parámetros de entrada (GET/POST)

  [pid=<pid>,usuario=<usuario>,pass=<pass>,nombre<nombre>,activo=<activo>,rol=<rol>,organizations<organizations>]

  El parámetro organizations, es un string seprados por “;” de las idOrganization

- FullURL:

http://91.126.141.4/users/alta?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceid=B4200044F0A0&usuario=<usuario>&pass=<pass>&nombre<nombre>&activo=<activo>&rol=<rol>&organizations<organizations>

❖ Parámetros de salida:

- Curso normal:

```
{
    "status": "ok",
    "message": OK",
}
```

Response code: 200

- Curso alternativo:

Pid incorrecto:

```
{
        "status": "ko",
        "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

Falta información (Usuario o Contraseña vacia):

```
{
        "status": "ko",
        "message": "INFORMATION_INCORRECT"
```

```
}
```

Response code: 400

Usuario existente:

```
{
    "status": "ko",
    "message": "USER_ALREDY_EXIST"
}
```

Response code: 400

❖ *Eliminar usuario*

- ❖ Descripción:

  Baja del usuario en la página web.

- ❖ URL:

  http://91.126.141.4/users/baja

- ❖ Parámetros de entrada (GET/POST)

  [pid=<pid>, usuario=<usuario>]

  - FullURL:

    http://91.126.141.4/users/baja?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&usuario=<usuario>

- ❖ Parámetros de salida:

  - Curso normal:

{

"status": "ok",

"message": OK",

}

Response code: 200

- Curso alternativo:

Pid incorrecto:

{

"status": "ko",

"message": "REQUEST_NOT_ALLOWED"

}

Response code: 400

Usuario existente:

{

"status": "ko",

"message": "USER_NOT_EXIST"

}

Response code: 400

❖ *Modificar de usuario*

- ❖ Descripción:

Actualizar usuario de la página web para un usuario.

- URL:

  http://91.126.141.4/users/modificar

- Parámetros de entrada (GET/POST)

  [pid=<pid>,usuario=<usuario>,pass=<pass>,nombre<nombre>,activo=<activo>,rol=<rol>, organizations =<organizations>](El usuario es obligatorio, los otros parámetros son opcionales)(El parámetro organizations, es un string seprados por ";" de las idOrganization)

  - FullURL:

    http://91.126.141.4/users/modificar?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceid=B4200044F0A0&usuario=<usuario>&pass=<pass>&nombre<nombre>&activo=<activo>&rol=<rol>

- Parámetros de salida:
  - Curso normal:

    {
    "status": "ok",
    "message": OK",
    }

    Response code: 200

  - Curso alternativo:

    Pid incorrecto:

    {

"status": "ko",
"message": "REQUEST_NOT_ALLOWED"
}
Response code: 400

Usuario inexistente:

{
"status": "ko",
"message": "USER_NOT_EXIST"
}
Response code: 400

Gestión de Clientes

❖ *Añadir Cliente*

- ❖ Descripción:

  Dar de alta a un cliente a la empresa Britel.

- ❖ URL:

  http://91.126.141.4/organization/alta

- ❖ Parámetros de entrada (GET/POST)

  [pid=<pid>,pais=<pais>,region=<region>,códigopostal=<códigopostal>,ciudad=< ciudad >,telefono = < telefono >, email = < email >]

  - FullURL:

    http://91.126.141.4/organization/alta?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceid=B4200044F0A0&organizacionid=123456789&active=1

- ❖ Parámetros de salida:

- Curso normal:

```
{
"status": "ok",
"message": OK",
}
```

Response code: 200

- Curso alternativo:

Pid incorrecto:

```
{
"status": "ko",
"message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

Acción no permitida:

```
{
"status": "ko",
"message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Eliminar Cliente*

- ❖ Descripción

  Dar de baja un cliente de Britel en el sistema.

- ❖ URL:

  http://91.126.141.4/organization/baja

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, organizationid = < organizationid >]

  - organizationid:

    id de organización en Britel (único)

  - FullURL:

    http://91.126.141.4/organization/baja?pid=971b4a1e83a20fef931

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

Organización inexistente:

```
{
    "status": "ko",
    "message": " ORGANIZATION_NOT_EXIST "
}
```

Response code: 400

acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Modificar Cliente*

- ❖ Descripción:

  Actualizar los datos de un cliente de la empresa Britel.

- ❖ URL:

  http://91.126.141.4/organization/modificar

- ❖ Parametros de entrada (GET/POST)

  [pid=<pid>,nombre<nombre>,pais=<pais>,region=<region>,dirección<direccion>, códigopostal = < códigopostal >, ciudad = < ciudad >, telefono =<telefono>, email = < email >]

  - FullURL:

    http://91.126.141.4/ organization/modificar?pid=971b4a1e83a20fef931b6

- ❖ Parametros de salida:

  - Curso normal:

    ```
    {
        "status": "ok",
    "message": OK",
    }
    ```

    Response code: 200

  - Curso alternativo:

    - Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```


Response code: 400

- Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Consultar Cliente de Usuario*

❖ Descripción:

Actualizar los datos de un cliente de la empresa Britel.

❖ URL:

http://91.126.141.4/organization/obtener

❖ Parámetros de entrada (GET/POST)

[pid=<pid>,organizacionId=<organizacionId>,pais=<pais>,region=<region>,códigoposta l=<codigopostal>,ciudad=<ciudad>,telefono=<telefono>,email=<email>]

- FullURL:

http://91.126.141.4/ organization/obtener?pid=971b4a1e83a20fef931b6

❖ Parámetros de salida:

- Curso normal:

```
{
    "status": "ok",
    "message": OK",
}
```

Response code: 200

- Curso alternativo:

Pid incorrecto:

```
{
        "status": "ko",
        "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

Acción no permitida:

```
{
        "status": "ko",
        "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

Gestionar Dispositivos

❖ *Añadir dispositivo*

- ❖ Descripción

  Añadir un dispositivo en el sistema.

- ❖ URL:

  http://91.126.141.4/suscriptors/alta

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>,deviceId=<deviceId>,active=<active>,organizationId<organizationId>,appid=<appid>]

  - deviceId:

    id de suscriptor en Britel (único)

  - FullURL:

    http://91.126.141.4/suscriptors/alta?pid=971b4a1e83a20fef931?pid=<pid>&deviceId = deviceId>&active=<active>&organizationId<organizationId>&appid<appid>

- ❖ Parámetros de Salida:

  - Curso normal:


```
{
    "status": "ok",
```

"message": OK"

}

Response code: 200

- Curso alternativo

pid incorrecto:

{

"status": "ko",

"message": "REQUEST_NOT_ALLOWED"

}

Response code: 400

Acción no permitida:

{

"status": "ko",

"message": "ACTION_NOT_ALLOWED"

}

Response code: 400

Introducir algún DeviceId:

{

```
"status": "ko",
"message": "INTRODUCE_DEVICEID"
}
```

Response code: 400

No existe la organizacion:

```
{
"status": "ko",
"message": “ORGANIZATION_NOT_EXIST "
}
```

Response code: 400

No existe ninguna app asociada:

```
{
"status": "ko",
"message": "APP_NOT_EXIST "
}
```

Response code: 400

❖ *Eliminar dispositivo*

- ❖ Descripción

  Eliminar un usuario en el sistema.

- ❖ URL:
  http://91.126.141.4/ suscriptors/baja

- ❖ Parámetros de Entrada (GET / POST):

[pid=<pid>, deviceId = <deviceId>]

- deviceId:

  id de suscriptor en Britel (único)

- FullURL:

  http://91.126.141.4/api/suscriptors/baja?pid=971b4a1e83a20fef931

Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": OK"
}
```

Response code: 200

- Curso alternativo

  Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id usuario inexistente:

```
{
    "status": "ko",
    "message": "STB_NOT_EXIST"
}
```


Response code: 400

Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

Introducir algun DeviceId:


```
{
    "status": "ko",
    "message": "INTRODUCE_DEVICEID"
}
```


Response code: 400

❖ *Modificar dispositivo*

- ❖ Descripción

Eliminar un usuario en el sistema.

- URL:

  http://91.126.141.4/ suscriptors/modificar

- Parámetros de Entrada (GET / POST):

  [pid=<pid>,deviceId=<deviceId>,active=<active>,organizationId<organizationId>,appid=<appid>]

  - deviceId:

    id de suscriptor en Britel (único)

  - FullURL:

    http://91.126.141.4/suscriptors/modificar?pid=971b4a1e83a20fef931&deviceId=<deviceId>&active=<active>&organizationId<organizationId>&appid=<appid>

- Parámetros de Salida:

  - Curso normal:

    ```
    {
        "status": "ok",
        "message": OK"
    }
    ```

    Response code: 200

  - Curso alternativo

    Pid incorrecto:

    {

```
"status": "ko",
"message": "REQUEST_NOT_ALLOWED"
}
```


Response code: 400

id usuario inexistente:


```
{
"status": "ko",
"message": "STB_NOT_EXIST"
}
```


Response code: 400

Acción no permitida:


```
{
"status": "ko",
"message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Consultar Dispositivos del Cliente*

- ❖ Descripción:

  Consultar los dispositivos asociados a un cliente.

- ❖ URL:

http://91.126.141.4/ suscriptors/obtener

- Parámetros de Entrada (GET / POST):

  [pid=<pid>, organizationId=< organizationId >]

  - deviceId:

    id de suscriptor en Britel (único)

  - organizationId:

    id del cliente de Birtel (único)

  - FullURL:

    http://91.126.141.4/suscriptors/obtener?pid=971b4a1e83a20fef931

- Parámetros de Salida:

  - Curso normal:

```
{
    "idDevice": "5as4dfa65":
    {
        "Active": “1"
    }
}
```

    Response code: 200

  - Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id organizacion no existe:

```
{
    "status": "ko",
    "message": "ORGANIZATION_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

Gestionar Parrillas

❖ *Eliminar Parrilla*

- ❖ Descripción:

  Elimina todos los canales configurados para la parrilla.

- ❖ URL:

  http://91.126.141.4/parrilla/baja

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, parrillaId=< parrillaId >]

  - parrillaId:

    id parrilla del cliente de Birtel (único)

  - FullURL:

  http://91.126.141.4/api/parrilla/baja?pid=971b4a1e83a20fef931&parrillaId=<parrillaId >

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id parrilla inexistente:

```
{
    "status": "ko",
    "message": "PARRILLA_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

Introducir algún DeviceId:


```
{
    "status": "ko",
    "message": "INTRODUCE_PARRILLAID"
}
```


Response code: 400

❖ *Modificar Parrilla*

- ❖ Descripción:

  Actualizar la parrilla configurada para el cliente.

- ❖ URL:

  http://91.126.141.4/parrila/modificar

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>,idparrilla=<idparrilla>,channelses<channelses>,paqueteses<paqueteses>]

  - organizationId:

    id del cliente de Birtel (único)

  - FullURL:

    http://91.126.141.4/api/parrilla/modificar?pid=971b4a1e83a20fef931

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id parrilla inexistente:

```
{
    "status": "ko",
    "message": "PARRILLA_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

Introducir algun ParrillaId:


```
{
    "status": "ko",
    "message": "INTRODUCE_PARRILLAID"
}
```


Response code: 400

❖ *Consultar Parrilla de un Cliente*

- Descripción:

  Consultar la parrilla configurada de cliente.

- URL:

  http://91.126.141.4/parrilla/obtener

- Parámetros de Entrada (GET / POST):

  [pid=<pid>, organizationId=< organizationId >]

- organizationId:

  id del cliente de Birtel (único)

- FullURL:

  http://91.126.141.4/api/parrilla/obtener?pid=971b4a1e83a20fef931

❖ Parámetros de Salida:

- Curso normal:

```
{
    “IdParrilla”:”654650”
        {
            Channels:
            {
                Nombre:
                {
                    AspectRatio:”4:3”
                    LogicalChannelNumber:”0”
                    …
                }
            }

            Paquetes:
            {
                NombrePaquete:
                {
                    Channels:
                    {
                        Nombre:
                        {
```

AspectRatio:”4:3”
LogicalChannelNumb
er:”0”
…
}
}
}
}
}
}

Response code: 200

- Curso alternativo

Pid incorrecto:

{
"status": "ko",
"message": "REQUEST_NOT_ALLOWED"
}

Response code: 400

id organización inexistente:

{
"status": "ko",
"message": "ORGANIZATION_NOT_EXIST"
}

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

Introducir algún ParrillaId:

```
{
    "status": "ko",
    "message": "INTRODUCE_ORGANIZATIONID"
}
```

Response code: 400

Gestionar Aplicaciones

❖ *Añadir Aplicación*

- Descripción:

  Eliminar una aplicación del sistema.

- URL:

  http://91.126.141.4/app/alta

❖ Parámetros de Entrada (GET / POST):

[pid=<pid>,organizationId=<organizationId>,nombre<nombre>,versión<version>,ubicacionServer< ubicacionServer >,suscriptores<suscriptores >]

- organizationId:

  id organization Birtel (único)

- FullURL:

http://91.126.141.4//app/alta?pid=971b4a1e83a20fef931[pid=<pid>,organizationId=<organizationId>,nombre<nombre>,versión<version>,ubicacionServer<ubicacionServer >]

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “ACTIVE"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
```

}

Response code: 400

id usuario existente:

{
"status": "ko",
"message": "APP_ALREDY_EXIST"
}

Response code: 400

id organización inexistente:

{
"status": "ko",
"message": " ORGANIZATION_NOT_EXIS"
}

Response code: 400

Acción no permitida:

{
"status": "ko",
"message": "ACTION_NOT_ALLOWED"
}

Response code: 400

❖ *Eliminar Aplicación*

- ❖ Descripción:

  Eliminar una aplicación del sistema.

- ❖ URL:

  http://91.126.141.4/app/baja

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, appid=< appid >]

  - FullURL:

    http://91.126.141.4//app/baja?pid=971b4a1e83a20fef931b68bff6fae242b4afce033356484a6b28d51a9f03cfc6&deviceId=B4200044F0A0

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

    Response code: 200

  - Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id app inexistente:

```
{
    "status": "ko",
    "message": "APP_NOT_EXIST"
}
```

Response code: 400

No se ha introducido ningún id app:

```
{
    "status": "ko",
    "message": " INTRODUCE_APPID"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Modificar Aplicación*

- ❖ Descripción:

  Actualizar la información de las aplicaciones.

- ❖ URL:

  http://91.126.141.4/app/modificar

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>,organizationId=<organizationId>,nombre<nombre>,versión<version>,ubicacionServer< ubicacionServer >]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

    http://91.126.141.4/app/modificar?pid=971b4a1e83a20fef931&pid=<pid>&organizationId=<organizationId>&nombre<nombre>&versión<version>&ubicacionServer< ubicacionServer >]

- Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id app inexistente:

```
{
    "status": "ko",
    "message": "APP_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Consultar Aplicación del cliente*

- ❖ Descripción:

  Eliminar la parrilla configurada para el cliente.

- ❖ URL:

  http://91.126.141.4/app/obtener

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, organizationId=< organizationId >]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

    http://91.126.141.4/app/obtener?pid=971b4a1e83a20fef931

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “ACTIVE"
}
```


Response code: 200

- Curso alternativo

Pid incorrecto:


```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```


Response code: 400

id usuario inexistente:


```
{
    "status": "ko",
    "message": "STB_NOT_EXIST"
}
```


Response code: 400

id usuario inactivo:


```
{
```

```
"status": "ko",
"message": "STB_NOT_ACTIVE"
}
```


Response code: 400

Acción no permitida:


```
{
"status": "ko",
"message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

Introducir algun DeviceId:


```
{
"status": "ko",
"message": "INTRODUCE_DEVICEID"
}
```


Response code: 400

Gestionar Canales

❖ *Añadir Canal*

- ❖ Descripción:

Añadir un canal al sistema.

- URL:

  http://91.126.141.4/channel/alta

- Parámetros de Entrada (GET / POST):

  [pid=<pid>,logicalChannelNumber=<logicalChannelNumber>,url=<url>,name=<name>, description=<description>,aspectRatio=<aspectRatio>,quality=<quality>,visibility=<visibility>,smallIcon=<smallIcon>,mediumIcon=<mediumIcon>,largeIcon=< largeIcon>, paqueteses< paqueteses>]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

http://91.126.141.4/channel/alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=<logicalChannelNumber>&url=<url>&name=<name>&description=<description>&aspectRatio=<aspectRatio>&quality=<quality>&visibility=<visibility>&smallIcon=<smallIcon>&mediumIcon=<mediumIcon>&largeIcon=<largeIcon>&paqueteses< paqueteses>

- Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

  Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

  Response code: 400

  Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

  Response code: 400

❖ *Eliminar Canal*

❖ Descripción:

Eliminar canal del sistema.

❖ URL:

http://91.126.141.4/channel/baja

❖ Parámetros de Entrada (GET / POST):

[pid=<pid>, idchannel=< idchannel >]

- FullURL:

http://91.126.141.4/channel/baja?pid=971b4a1e83a20fef931&pid=<pid>&idchannel=< idchannel >

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel no introducido:

```
{
    "status": "ko",
    "message": " INTRODUCE_CHANNELID"
}
```


Response code: 400

id channel inexistente:


```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```


Response code: 400

Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Modificar Canal*

- ❖ Descripción:

  Modificar canal del sistema.

- URL:

  http://91.126.141.4/channel/modificar

- Parámetros de Entrada (GET / POST):

  [pid=<pid>,logicalChannelNumber=<logicalChannelNumber>,url=<url>,name=<name>, description=<description>,aspectRatio=<aspectRatio>,quality=<quality>,visibility=<visibility>,smallIcon=<smallIcon>,mediumIcon=<mediumIcon>,largeIcon=< largeIcon>, paqueteses< paqueteses>,idchannels< idchannels >]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

  http://91.126.141.4/channel/alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=<logicalChannelNumber>&url=<url>&name=<name>&description=<description>&aspectRatio=<aspectRatio>&quality=<quality>&visibility=<visibility>&smallIcon=<smallIcon>&mediumIcon=<mediumIcon>&largeIcon=< largeIcon>&paqueteses< paqueteses>&idchannels< idchannels >

- Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Consultar información de un canal*

- ❖ Descripción:

  Modificar canal del sistema.

- ❖ URL:

  http://91.126.141.4/channel/obtener

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, logicalChannelNumber=< logicalChannelNumber >, url=<url>, name=<name>, description=< description>, aspectRatio=< aspectRatio>, quality=< quality>, visibility=< visibility>, smallIcon=< smallIcon>, mediumIcon=< mediumIcon>, largeIcon=< largeIcon>, paqueteses< paqueteses>,idchannels< idchannels >]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

  http://91.126.141.4/channel/alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=<logicalChannelNumber>&url=<url>&name=<name>&description=<description>&aspectRatio=<aspectRatio>&quality=<quality>&visibility=<visibility>&smallIcon=<smallIcon>&mediumIcon=<mediumIcon>&largeIcon=< largeIcon>&paqueteses< paqueteses>&idchannels< idchannels >

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
```

"message": "ACTION_NOT_ALLOWED"

}

Response code: 400

Gestionar Paquetes

❖ *Añadir Paquete*

- ❖ Descripción:

  Añadir un paquete al sistema.

- ❖ URL:

  http://91.126.141.4/paquetes/alta

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>,]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

http://91.126.141.4/channel /alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=< logicalChannelNumber >&url=<url>&name=<name>&description=<description>&aspectRatio=< aspectRatio>&quality=< quality>&visibility=< visibility>&smallIcon=< smallIcon>& mediumIcon=< mediumIcon>& largeIcon=< largeIcon>&paqueteses< paqueteses>

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Eliminar Paquete*

- ❖ Descripción:

  Eliminar canal del sistema.

- ❖ URL:

  http://91.126.141.4/paquetes/baja

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, idchannel=< idchannel >]

  - FullURL:

http://91.126.141.4/channel/baja?pid=971b4a1e83a20fef931&pid=<pid>&idchannel=< idchannel >

- ❖ Parámetros de Salida:

  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

  - Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel no introducido:

```
{
    "status": "ko",
    "message": " INTRODUCE_CHANNELID"
}
```

Response code: 400

id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
```

"status": "ko",
"message": "ACTION_NOT_ALLOWED"
}

Response code: 400

❖ *Modificar Paquete*

- ❖ Descripción:

  Modificar paquete del sistema.

- ❖ URL:

  http://91.126.141.4/paquetes/modificar

- ❖ Parámetros de Entrada (GET / POST):

  [pid=<pid>, logicalChannelNumber=< logicalChannelNumber >, url=<url>, name=<name>, description=< description>, aspectRatio=< aspectRatio>, quality=< quality>, visibility=< visibility>, smallIcon=< smallIcon>, mediumIcon=< mediumIcon>, largeIcon=< largeIcon>, paqueteses< paqueteses>,idchannels< idchannels >]

  - organizationId:

    id organization Birtel (único)

  - FullURL:

    http://91.126.141.4/channel /alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=< logicalChannelNumber >&url=<url>&name=<name>&description=<description>&aspectRatio=< aspectRatio>&quality=< quality>&visibility=< visibility>&

smallIcon=< smallIcon>& mediumIcon=< mediumIcon>& largeIcon=< largeIcon>&paqueteses< paqueteses>&idchannels< idchannels >

- ❖ Parámetros de Salida:
  - Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

  - Curso alternativo
    - pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

    - id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```

Response code: 400

- acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400

❖ *Obtener información de un paquete*

❖ Descripción:

Modificar paquete del sistema.

❖ URL:

http://91.126.141.4/paquetes/modificar

❖ Parámetros de Entrada (GET / POST):

[pid=<pid>, logicalChannelNumber=< logicalChannelNumber >, url=<url>, name=<name>, description=< description>, aspectRatio=< aspectRatio>, quality=< quality>, visibility=< visibility>, smallIcon=< smallIcon>, mediumIcon=< mediumIcon>, largeIcon=< largeIcon>, paqueteses< paqueteses>,idchannels< idchannels >]

- organizationId:

id organization Birtel (único)

- FullURL:

http://91.126.141.4/channel /alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=< logicalChannelNumber >&url=<url>&name=<name>&description=<description>&aspectRatio=< aspectRatio>&quality=< quality>&visibility=< visibility>& smallIcon=< smallIcon>& mediumIcon=< mediumIcon>& largeIcon=< largeIcon>&paqueteses< paqueteses>&idchannels< idchannels >

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```


Response code: 400

Acción no permitida:


```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```


Response code: 400

❖ *Obtener canales de un paquete*

- Descripción:

  Modificar paquete del sistema.

- URL:

  http://91.126.141.4/paquetes/obtenerCanales

- Parámetros de Entrada (GET / POST):

  [pid=<pid>, logicalChannelNumber=< logicalChannelNumber >, url=<url>, name=<name>, description=< description>, aspectRatio=< aspectRatio>, quality=< quality>, visibility=< visibility>, smallIcon=< smallIcon>, mediumIcon=< mediumIcon>, largeIcon=< largeIcon>, paqueteses< paqueteses>,idchannels< idchannels >]

- organizationId:

  id organization Birtel (único)

- FullURL:

http://91.126.141.4/channel/alta?pid=971b4a1e83a20fef93&pid=<pid>&logicalChannelNumber=<logicalChannelNumber>&url=<url>&name=<name>&description=<description>&aspectRatio=<aspectRatio>&quality=<quality>&visibility=<visibility>&smallIcon=<smallIcon>&mediumIcon=<mediumIcon>&largeIcon=< largeIcon>&paqueteses<paqueteses>&idchannels< idchannels >

❖ Parámetros de Salida:

- Curso normal:

```
{
    "status": "ok",
    "message": “OK"
}
```

Response code: 200

- Curso alternativo

Pid incorrecto:

```
{
    "status": "ko",
    "message": "REQUEST_NOT_ALLOWED"
}
```

Response code: 400

id channel inexistente:

```
{
    "status": "ko",
    "message": " CHANNEL_NOT_EXIST"
}
```

Response code: 400

Acción no permitida:

```
{
    "status": "ko",
    "message": "ACTION_NOT_ALLOWED"
}
```

Response code: 400